



US006055516A

United States Patent [19]
Johnson et al.**[11] Patent Number: 6,055,516**
[45] Date of Patent: Apr. 25, 2000**[54] ELECTRONIC SOURCING SYSTEM****[75] Inventors:** James M. Johnson, Bridgeville;
Robert P. Kinross, Ben Avon; Francis
J. Melly, Pittsburgh; Douglas A.
Momyer, Upper St. Clair, all of Pa.**[73] Assignee:** Procurenent, Inc., Fairfield, N.J.**[21] Appl. No.:** 09/234,366**[22] Filed:** Jan. 20, 1999**Related U.S. Application Data**

- [63]** Continuation of application No. 08/288,577, Aug. 10, 1994.
[51] Int. Cl.⁷ G06F 17/60
[52] U.S. Cl. 705/27; 705/26; 701/1;
701/10; 701/104
[58] Field of Search 705/1, 26, 27;
707/1, 10, 100, 104

[56] References Cited**U.S. PATENT DOCUMENTS**

4,135,241	1/1979	Stanis et al.	364/200
4,141,078	2/1979	Bridges, Jr. et al.	364/900
4,336,589	6/1982	Smith et al.	364/403
4,509,123	4/1985	Vereen	364/300
4,636,950	1/1987	Caswell et al.	364/403
4,656,591	4/1987	Goldberg	364/478
4,887,208	12/1989	Schneider et al.	364/403
4,897,867	1/1990	Foster et al.	379/94
4,920,488	4/1990	Filley	364/403
4,958,280	9/1990	Pauly et al.	364/403
4,972,318	11/1990	Brown et al.	364/403
4,984,155	1/1991	Geier et al.	364/401
4,992,940	2/1991	Dworkin	364/401
5,038,283	8/1991	Caveney	364/403
5,077,665	12/1991	Silverman et al.	364/408
5,101,352	3/1992	Rembert	364/401
5,103,079	4/1992	Barakai et al.	235/380
5,117,354	5/1992	Long et al.	364/401
5,168,444	12/1992	Cukor et al.	364/401
5,168,445	12/1992	Kawashima et al.	364/403
5,305,199	4/1994	LoBiondo et al.	364/403
5,319,542	6/1994	King, Jr. et al.	364/401

5,758,327	5/1998	Gardner et al.	705/26
5,778,355	7/1998	Boyer et al.	707/2
5,799,289	8/1998	Gukushima et al.	705/400

FOREIGN PATENT DOCUMENTS

0 697 669 A2 8/1995 European Pat. Off.
WO 90/11572 10/1990 WIPO

OTHER PUBLICATIONS

European Search Report, Application No. 95 305364, Dated Aug. 2, 1997.

"Texas Instruments Puts the MRO Buy On Line," *Purchasing*, p. 37, May 19, 1994.

"IBM Technical Viewer/2 General Information Manual," IBM Corporation, 1991.

"IBM Technical Viewer/2" product information brochure, IBM Corporation, undated.

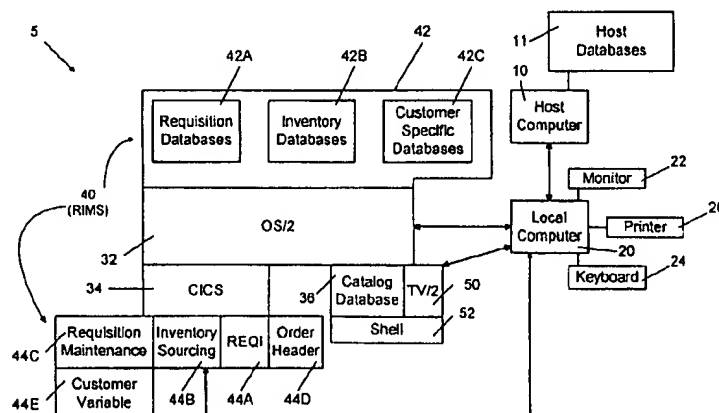
(List continued on next page.)

Primary Examiner—Edward R. Cosimano

Attorney, Agent, or Firm—Rader, Fishman & Grauer PLLC

[57] ABSTRACT

An electronic sourcing system includes a computer that maintains a catalog database of data including product information (such as product identification and descriptive information) relating to catalog items available from vendor product catalogs, and a means for building (generating) a requisition including at least one requisitioned item. Information at least partially identifying an item desired to be requisitioned is entered by a user, and utilized by a means for searching the database for catalog items matching that information and for selecting at least one catalog item located as a result of the search. Text describing the catalog items, and images of the items, may be viewed. Data identifying selected catalog items are communicated to the requisition building means, which generates a requisition including entries for items corresponding to the selected catalog items. The system checks the availability in one or more inventory locations of the corresponding desired catalog items, and generates one or more purchase orders for desired items from inventory locations stocking the items.

29 Claims, 5 Drawing Sheets

OTHER PUBLICATIONS

"Guide to Using First Place," Automated Catalogue Services Inc., Aug. 1993.

"Grainger Electronic Catalog User's Guide," Edition 6, pp. 29-33 and 49-50, 1994.

"Automated Catalog Systems" product information brochure, Distrivision Development Corporation, 1994.

"SweetSource User's Guide," Sweet's Electronic Publishing, Mar. 1, 1993.

"Aldrich Catalog on Disk Reference Manual," Aldrich Chemical Company, Inc., 1994.

"Easel to Move Development to Windows," *Infoworld*, May 16, 1994.

"Easel Eases the Way to OO," *Computer Select*, Jun. 1994.

"Stockclerk Inventory Management System" product information brochure, IOBar, Inc., 1993.

"Fisher StockPro™ Inventory Management System" brochure, Fisher Scientific, 1990.

"Systems by Fisher/Inventory Management System/StockPro®—Single-User Version" user manual, pp. 6-1 to 6-14, 6-21 to 6-26, 7-1 to 7-18, 8-1 to 8-20, 9-1 to 9-6, and 10-1 to 10-8, Fisher Scientific Company, Feb. 28, 1989.

"Fisher Scientific PurchasePro" user manual, version 1.1., Introduction pp. 1-6, File Editor pp. 1-12, Purchase Requisitions pp. 1-12, Purchase Orders pp. 1-8, 23, and 34, Vendor Quotes p. 1, Fastback Orders pp. 1-8, Reports p. 1, ReportPro p. 1, 1984.

"Lightning™ Fisher's Electronic Order Entry System" brochure, Fisher Scientific Company, 1989.

"Lightning™ Order Entry and Information System" user manual, pp. 1-6, 19-54, and 91-96, Fisher Scientific Company, 1990.

"Fisher Reliance™ System" brochure, Fisher Scientific Company, 1989.

"Fisher 88" catalog, pp. 1536, 1549-1567, Fisher Scientific Company, 1987.

Excerpt from "A Prism Electronic Catalog Proposal: Product Information System for Sales and Marketing," prepared by Fisher Scientific Company, presented by MediaShare Corporation, Aug. 31, 1993.

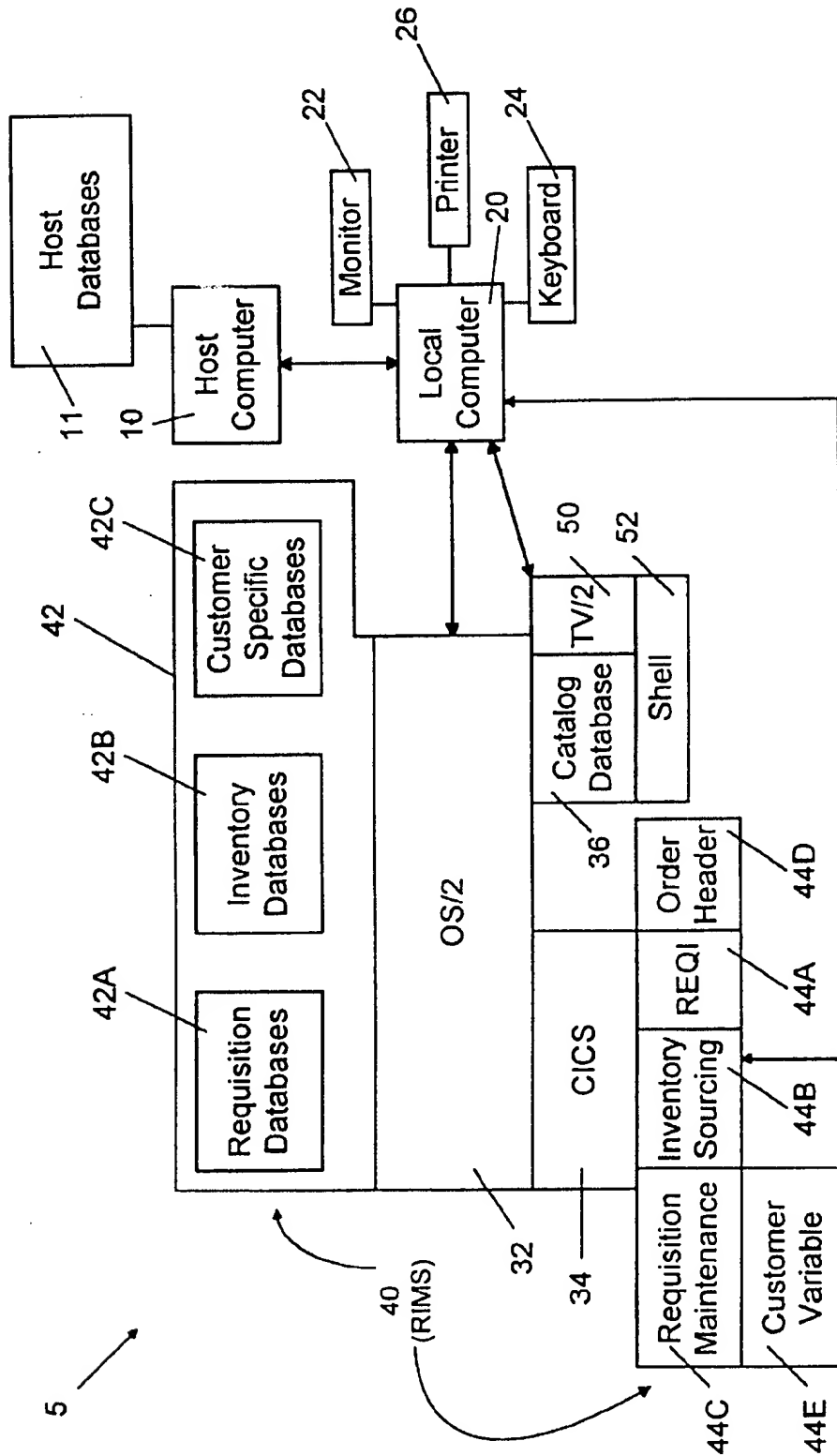


FIG. 1A

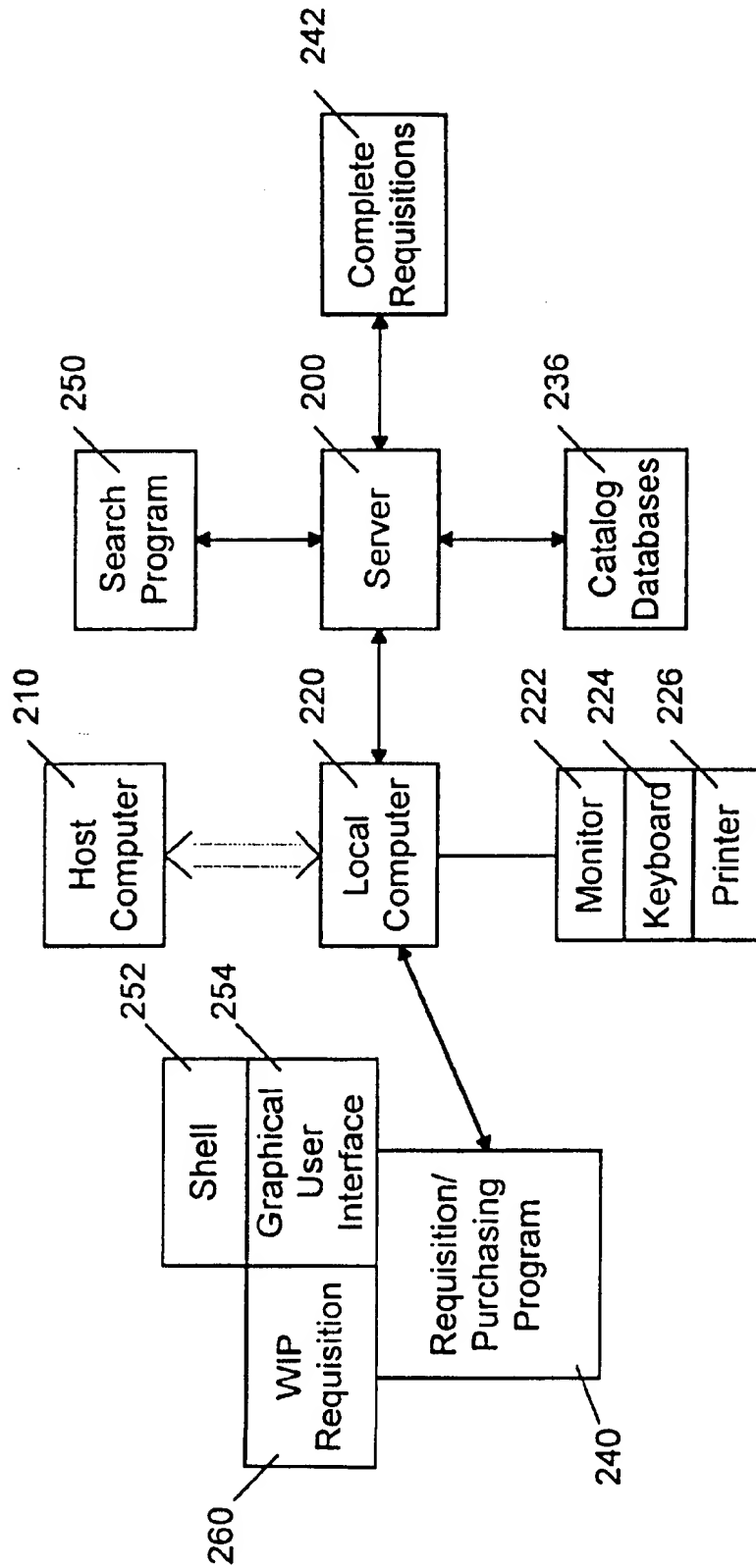


FIG. 1B

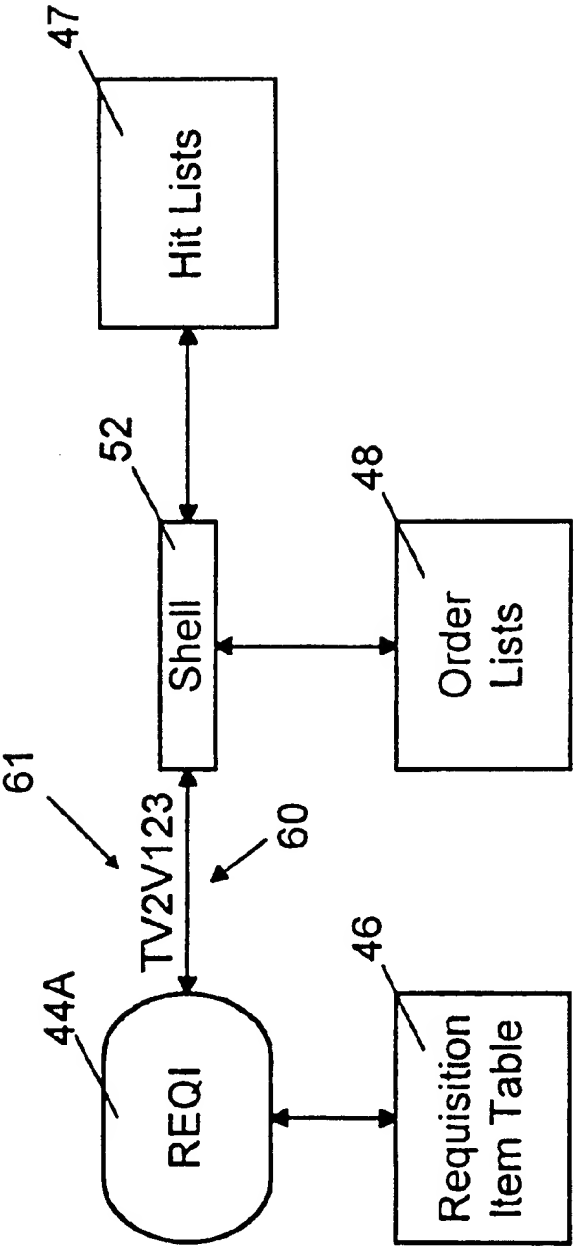


FIG. 1C

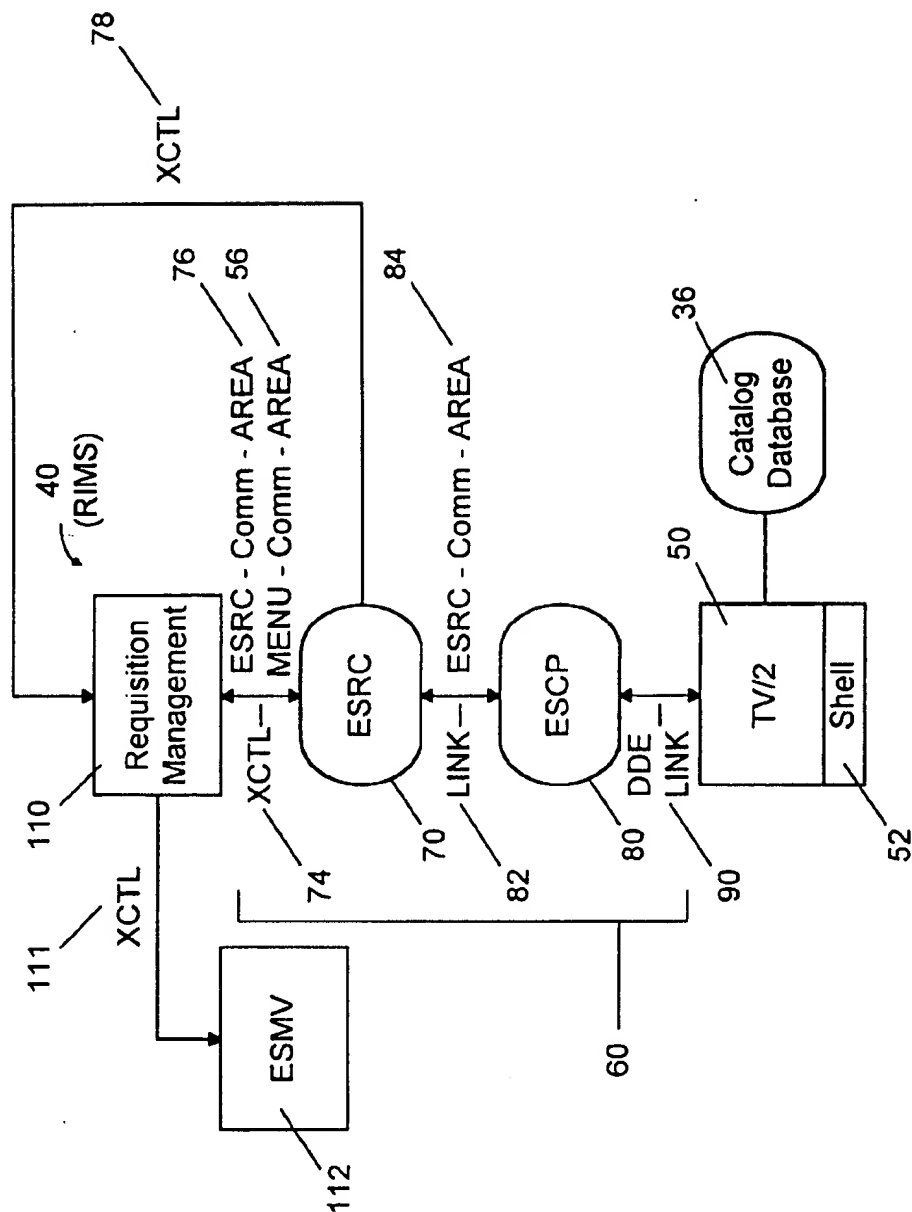


FIG. 2

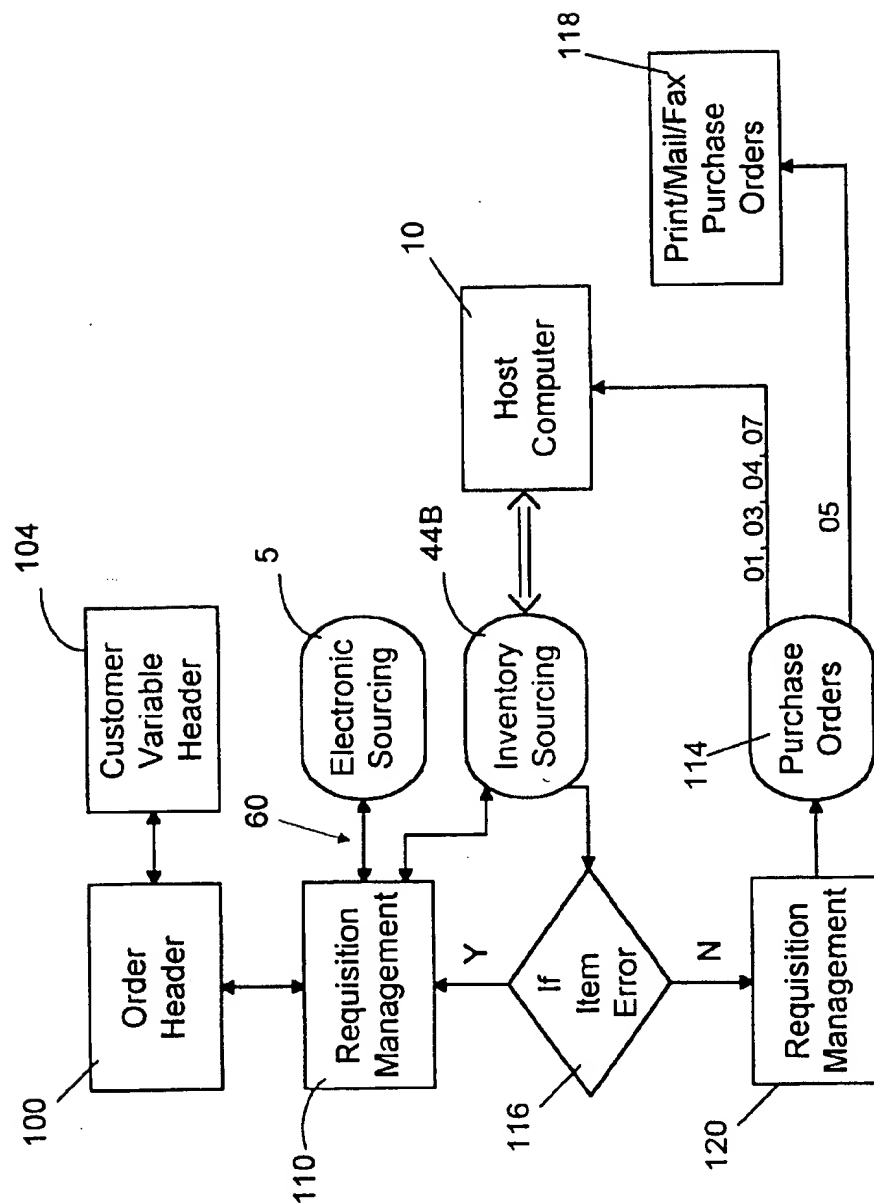


FIG. 3

ELECTRONIC SOURCING SYSTEM

This application is a continuation of U.S. application Ser. No. 08/288,577 filed on Aug. 10, 1994.

BACKGROUND OF THE INVENTION

This invention relates to systems and methods for interfacing product information, such as is typically found in vendor catalogs that are provided to customers, and requisition/purchasing systems and methods that may use the results of searches of product information.

There are a number of known requisition/purchasing systems that manage and process requisitions and purchase orders. One such system is the Fisher Scientific Requisition and Inventory Management System ("Fisher RIMS"), described in co-pending patent application Ser. No. 08/042,168, filed Apr. 2, 1993, issued as U.S. Pat. No. 5,712,989 on Jan. 28, 1998, and assigned to Fisher Scientific Company of Pittsburgh, Pa., the disclosure of which is incorporated herein by reference. As its title suggests, Fisher RIMS can also manage inventory. In the Fisher RIMS system, requisition records are created from a real-time interaction between a host computer (generally a mainframe) and a local computer (generally at a customer site), with each computer using data from its own respective database of inventory in conjunction with information entered by a customer service representative operating the local computer. By accessing its respective database, each computer can build and transmit to the other computer communications blocks of data relating to a particular requisition of an item in inventory (or to the management of the inventory itself). The other computer can then use the received data to continue processing of the requisition. Thus, requisition records are created from a real-time interaction between the host and local computers, with each computer using data from its respective database in conjunction with information entered by a customer service representative operating the local computer.

Other requisition/purchasing systems can be grouped broadly into four classes. First, requisition management systems licensed to corporations purchasing for their own use include ORION software (from Medical Management Systems), ENTERPRISE software (from ESI), and NOVA software (from Johnson & Johnson). Second, there exist systems provided by distributors for transmitting orders to them in proprietary formats. Such systems include QUICK-LINK (from Abbott), ASAP system (from Baxter) and LIGHTNING system (from Fisher Scientific). Third, software packages licensed by software developers to customers and/or suppliers enable the transmission of customer purchase orders as EDI purchase orders (in ANSI X.12 format). Examples of such systems include ON-CALL EDI (from TSI International), EDI Express software (from General Electric Information Services) and GETRAN software (from Sterling Software). Fourth, comprehensive business management packages such as REAL WORLD software (from Real World Corporation of Concord, N.H.) and ASK software (from The ASK Group) contain a purchasing module to create replenishment orders when inventoried items fall below restocking points. The same purchasing module can also be used to place spot orders for products keyed in by the customer's purchasing personnel.

None of these known requisition/purchasing systems (including Fisher RIMS), however, provides a capability for a user readily to search for and locate information about the products that may be requisitioned and ordered in connec-

tion with the requisition/purchasing system. They also do not provide the capability for a user to search a database containing two or more vendor catalogs, and then to transfer information about the items selected as a result of such searches into a requisition/purchasing system such as Fisher RIMS for building a requisition for the catalog items.

Computer systems that are capable of searching databases containing a product catalog of a particular vendor, for example on CD-ROM, are also known. Such systems can search for user requested information about products and create orders which the user can save, print or, in some cases, facsimile directly to a vendor. The known computer systems for searching vendor catalogs are limited in that only one such vendor catalog is accessible to a user at any given time. They are also limited in that they can only create an order within the particular vendor catalog database. They cannot source items to be requisitioned from a database containing multiple catalogs or interact with a requisition/purchasing system (such as Fisher RIMS) to create a purchase order or orders including the items located from that sourcing operation.

Thus, it would be desirable to provide an electronic sourcing system that provides a means for transferring information between a requisition/purchasing system that may use the results of a search of product information and a means for searching large volumes of product information such as would be included in a vendor product catalog or catalogs.

It would also be desirable to provide such an electronic sourcing system that is capable of searching a database containing at least two vendor product catalogs for product information.

It would further be desirable to provide such an electronic sourcing system that is capable of searching a database of catalog items contain in at least two vendor product catalogs, selecting particular items located, and transferring information about the items selected (for example, a catalog number and a vendor identifier, such as vendor name and/or vendor number) to a requisition/purchasing system for inclusion in a requisition generated by the system.

It would further be desirable to provide an electronic sourcing system that is capable of creating an order list including items located as the result of a catalog database search and transferring that order list of desired catalog items to a requisition/purchasing system for inclusion of the catalog items as entries in a requisition generated by the system.

SUMMARY OF THE INVENTION

In view of the foregoing, it is an object of this invention to provide an electronic sourcing method and system that provides a user with the capability of searching a database containing data (including product/vendor identification, and other product information) relating to items available from at least two vendor product catalogs, and the capability of transferring the product information for desired catalog items obtained as a result of the search to a requisition/purchasing system for use in generating a requisition including entries for the desired catalog items.

It is also an object of this invention to provide an electronic sourcing system that provides a means for bi-directionally transferring information between a requisition/purchasing system that may use the results of a search of such product information, and a means for searching large volumes of product information such as would be included in a vendor product catalog.

3

It is a further object of this invention to provide an electronic sourcing system capable of creating an order list including desired catalog items located as the result of such a database search, and transferring that order list to a requisition/purchasing system for generating a requisition including entries for the desired catalog items.

In accordance with the invention, an electronic sourcing system and method used by the system are provided. The system includes a computer that maintains a catalog database of data including product information (such as product identification information, and descriptive information) relating to catalog items available from vendor product catalogs, and a means for building (generating) a requisition including at least one requisitioned item. Information at least partially identifying an item desired to be requisitioned is entered by a user, and utilized by a means for searching the database for catalog items matching that information and for selecting at least one catalog item located as a result of the search. Text describing the catalog items, and images of the items, may be viewed. Data identifying selected catalog items are communicated to the requisition building means, which generates a requisition including entries for items corresponding to the selected catalog items. Additionally, the invention includes a means for checking the availability in one or more inventory locations of the corresponding desired catalog items, and for generating one or more purchase orders for desired items from inventory locations stocking the items.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects and advantages of the invention will be apparent from consideration of the following detailed description, taken in conjunction with the accompanying drawings, in which like reference characters refer to like parts throughout, and in which:

FIG. 1A is a block diagram showing one exemplary embodiment of the overall system of the present invention;

FIG. 1B is a block diagram showing another exemplary embodiment of the overall system of the present invention;

FIG. 1C is a block diagram showing a portion of the embodiment of FIG. 1A in greater detail;

FIG. 2 is a block diagram showing the flow of control and interaction between the various programs and data screens of the programs used for requisition management and vendor catalog searching of the present invention; and

FIG. 3 is a block diagram showing a portion of a system (Fisher RIMS) for requisition management, including the electronic sourcing system of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

FIGS. 1A and 1B show preferred embodiments of the electronic sourcing system 5 of the present invention. As shown in FIG. 1A, a local computer 20, which is preferably located at or near a Customer site and the site of Just-In-Time ("JIT") Inventory, is preferably used by an on-site Customer Service Representative ("CSR") dedicated to a Customer to assist that Customer in requisitioning items needed.

Local computer 20 includes conventional color monitor 22 and alphanumeric keyboard 24 including twelve function keys F1, F2, . . . F12. Local computer 20 is also coupled to printer 26.

Local computer 20 is preferably a conventional micro-computer (such as a 386-, 486- or Pentium-class personal

4

computer) capable of operating the required programs and of transmitting and receiving the required communications, running the OS/2 operating system 32 and also running the CICS OS/2 application 34, both of which are available from IBM.

Electronic sourcing system 5 also includes a requisition/purchasing system 40, preferably but not necessarily the Fisher RIMS system, and a search program 50 that is capable of searching through large volumes of information quickly and accurately. Preferably but not necessarily, the Technical viewer 2 search program ("TV/2"), available from IBM, is used as search program 50. As shown in the embodiment of FIG. 1A, Fisher RIMS 40 and TV/2 search program 50 are run by local computer 20.

Fisher RIMS system 40 is comprised of numerous program modules, including several programs 44, which operate within CICS environment 34 of OS/2 operating system 32. Programs 44 include, among others, Requisition Management ("REQUI") program 44A, Inventory Sourcing program or programs 44B, Requisition Maintenance program 44C, Customer Variable program 44D, and Order Header program 44E, each of which will later be described in greater detail. REQUI program 44A is most often the RIMS program 44 that interfaces with TV/2 search program 50.

Fisher RIMS system 40 also includes several Fisher RIMS databases 42. These databases 42 preferably include requisition databases 42A, inventory databases 42B, and customer-specific databases 42C, each maintained within OS/2 operating system 32.

Local computer 20 also preferably runs Shell program 52, which operates under search program 50 and is used to customize search program 50 to generate Order Lists 48 (shown in FIG. 1C) with particular fields of formatted data about the items selected using search program 50. Local computer 20 is preferably capable of running both a RIMS program 44 and Shell program 52 at the same time (i.e., in a multi-tasking environment), but the user of local computer 20 usually sees only RIMS program 44 or Shell program 52 at one time in the foreground on monitor 22.

Local computer 20 is also provided with a catalog database 36 comprised preferably of at least two vendor product catalogs. The catalogs, and hence catalog database 36, preferably include such information as part number, price, catalog number, vendor name or I.D., and vendor catalog number, as well as textual information and images of or relating to the catalog products. The nature of the business that the Customer using electronic sourcing system 5 conducts will determine which product catalogs are made a part of catalog database 36.

A feature of the present invention is the ability to search multiple catalogs from different suppliers. For example, catalog database 36 can contain the catalog or catalogs published by a vendor Distributor, having Distributor's catalog numbers for all listed products and vendor manufacturer's part numbers for many of the listed products. Catalog database 36 can further contain catalogs published by some of the vendor manufacturers, listing the manufacturers' part numbers for certain products correspondingly listed in the Distributor's catalogs and for certain products not listed in the Distributor's catalogs. Catalog database 36 can further contain catalogs published by outside suppliers, whether other manufacturers or other distributors, listing such vendor's products different from those in the Distributor's catalogs.

Where the Fisher RIMS system is in use with electronic sourcing system 5, a host computer 10 located at a Distribu-

tor site is also provided, as shown in FIG. 1A. Host computer 10 controls all inventory, pricing and requisitioning operations of the Distributor's regularly stocked items using host pricing and inventory databases 11. Host pricing and inventory databases 11 may include such information as: descriptions of the items and the quantities thereof available at a particular Distributor warehouse and at other Distributor warehouses; item records for each Product regularly sold by the Distributor; discount records by Customer; and cross-references from the Distributor's catalog number to its corresponding vendor's part (catalog) number and to similar corresponding catalog numbers of other vendors (suppliers or distributors) for the same Product.

Host computer 10 and local computer 20 are preferably linked point-to-point or in a network employing the formats and protocols of IBM's System Network Architecture ("SNA"). Host computer 10 can be substantially any mainframe or minicomputer capable of running the desired programs and conducting the required communications. Preferably, host computer 10 is a mainframe computer, such as an IBM Model 3090, running the MVS operating system, the MVS-CICS application and a Virtual Telecommunication Access Method communications network.

As shown in FIGS. 1C and 2, interface 60 is also a part of electronic sourcing interface system 5. Interface 60 communicates shared data between requisition/purchasing system 40 and search program 50. Interface 60 is preferably based upon the dynamic data exchange ("DDE") protocol provided by OS/2 operating system 32. As shown in FIG. 2, interface 60 preferably includes three linking programs to interface requisition/purchasing system 40 and search program 50: ESRC program 70, ESCP program 80 and DDE LINK 90.

A typical data exchange may begin with requisition/purchasing system 40 (which, in the illustrated embodiment, is the Fisher RIMS system) requesting information from catalog database 36 via search program 50. Once a search by search program 50 has been completed, the selected information will be communicated to requisition/purchasing system 40 via interface 60.

Alternatively, if the search of catalog database 36 is initiated from search program 50, the information selected from the search is returned to requisition/procurement system 40 via interface 60.

The start up of electronic sourcing system 5 (FIG. 1A) may be user-initiated or automatically started when the operating system, preferably OS/2 system 32, is brought up on local computer 20. An application-name string 61 must be identified to label interface 60. As shown in FIG. 1C, electronic sourcing system 5 by convention will use "TV2V123," "TV2V124," "TV2V125," etc. as application names 61 supporting the user's requesting service.

Preferably, application names 61 correspond to virtual terminal sessions that exist in the CICS system 34 of requisition/purchasing system 40. There will be a one-to-one correspondence between applications started (such as Shell 52) and CICS virtual terminals in use at a location of requisition/procurement system 40 (such as REQI program 44A). Local computer 20 will query OS/2 operating system 32 to determine the next application-name string 61 to create at start-up. The application-name strings 61 will be created in sequence with V123 being created first, V124 created second, etc. Each application will create only one application name-string 61 to support its user in the CICS environment 34.

If the Fisher RIMS system has been selected as requisition/purchasing system 40, and the TV/2 search pro-

gram has been selected as search program 50, CICS OS/2 applications 34 must share a workstation with a TV/2 search program 50.

The data passed by interface 60 preferably comprise all or a subset of the following twelve fields: vendor name, vendor number, vendor part (catalog) number, product description, bid price, list price, keyword, page number, quantity, unit, catalog text, and catalog images. Because of the amount of data for catalog images present in database 36 and viewed on monitor 22, these data are usually not passed via interface 60. Any of the above-listed fields may be filled by requisition/purchasing system 40 prior to requesting a search of catalog database 36 by search program 50. However, requisition/purchasing system 40 is not required to pass any data to search program 50. If a field is not passed, that field will be filled with spaces. The fields that are filled with data will assist search program 50 in executing its first search against a specific catalog contained in catalog database 36.

A search priority exists when more than one field is provided by requisition/purchasing system 40. The priority is as follows: (1) part (catalog) number; (2) keyword; and (3) page number. The search will start with priority (1) and proceed through priority (3) in sequence until a search produces products matching the search criteria. At that time, the search will return the matching product information to requisition/purchasing system 40 and stop at the highest priority resulting in a match.

The operation of electronic sourcing system 5 of the present invention will now be more particularly described in the context of FIGS. 1A, 1C, 2 and 3. In FIGS. 2 and 3, the rectangles represent data screens as well as programs associated with those data screens. The rounded rectangles represent programs not associated with data screens such that, while these programs are running, the prior data screen may remain visible without, necessarily, being operational for the input of data. The programs associated with the data screens enable the user of local computer 20 to display and modify the contents of various tables associated with particular data screens. The following description illustrates the use of the Fisher RIMS system as requisition/purchasing system 40, and the TV/2 search program as search program 50. However, it will be understood that the present invention is not limited to such system or program.

Preferably, a user will start the electronic sourcing system 5 from Fisher RIMS system 40. Requisitioning on Fisher RIMS system 40 in context of the electronic sourcing system 5 of the present invention is illustrated in pertinent part in FIG. 3 (and is fully described in application Ser. No. 08/042,168, now U.S. Pat. No. 5,712,989). As data (e.g., Account Number, Requisition Number and Stock Numbers) associated with a single requisition are entered through the various data screens on local computer 20, that computer creates a set of Requisition Tables (including a Requisition Item Table 46, shown in FIG. 1C) for that particular requisition. The Requisition Tables are stored in Requisition databases 42A (shown in FIG. 1A), and can be accessed by local computer 20 using the Requisition Number to find the desired table.

The first step in creating a requisition in Fisher RIMS system 40 involves entry by the user of information in the Order Header program 44D (shown in FIG. 1A), which has an associated Order Header data screen 100 (FIG. 3). A sample of an actual Order Header data screen 100 is set forth in Appendix I. The user enters an Account Number, which generally causes the correct name and address associated with that Account Number to be entered into the appropriate

fields of Order Header data screen 100. The user must also enter a Requisition Number in the appropriate field of the Order Header screen 100. Various additional information may also be entered.

At the bottom of Order Header data screen 100 are several fields that describe the function of various function keys. Function keys F6, F9, and F10 all cause the system to jump to a new RIMS program 44 or data screen in Fisher RIMS system 40. For example, pressing the F9 key causes the system to jump to RIMS Customer Variable program 44E (FIG. 1A) and its associated Customer Variable Header data screen 104 (FIG. 3). Customer Variable Header program 44E with its associated Customer Variable Header data screen 104 allows the user to enter and edit information that the particular customer desires to be associated with the requisition due to requirements of the customer's internal accounting system or other systems. Pressing the F10 key will cause the system to enter the Inventory Sourcing program or programs 44B.

Pressing the F6 function key from the Order Header data screen causes Fisher RIMS system 40 to jump to REQI program 44A (FIG. 1A). The screen associated with REQI program 44A is Requisition Management data screen 110 (FIG. 3) illustrated in Appendix II. Within REQI program 44A and its associated Requisition Management data screen 110, Requisition Item Table 46 (shown in FIG. 1C) is a graphical representation of a database table in which certain fields are completed on a list of items that are to be listed, sourced and ordered. Representative Requisition Management data screens 110 showing a Requisition on Requisition Item Table 46 are set forth in Appendices II, VIII and IX. It should be appreciated that data about each item is stored in Requisition Item Table 46, some of which is displayed on the screens shown in Appendices II, VIII and IX. The data stored can additionally include customer variable data. That is, the fields on Requisition Item Table 46 can be expanded to include specific item details used by a particular customer, especially when reports from requisition databases are transferred to the customer's host computer (not shown). The field structure for these data is maintained in customer-specific databases 42C.

The entire process of listing, sourcing and ordering products using Fisher RIMS system 40 can be completed without any reference to a search program 50. As described herein, however, limited fields on specific items can be transmitted from Requisition Item Table 46 to search program 50, and more completed fields of the same or different items can be received from the search program 50 into a Requisition Item Table 46.

At the bottom of Requisition Management data screen 110 (FIG. 3), and Appendices II, VIII and IX) are several fields which describe the function of various function keys (F1, F2, etc.). The user uses REQI program 44A and its associated Requisition Management data screen 110 to enter the catalog or part numbers and quantities of the various items being requisitioned.

The Account Number and Requisition Number are automatically passed to REQI program 44A and its associated Requisition Management data screen 110, and displayed at the top of the Requisition Management data screen 110 in the relevant fields. For example, in the exemplary Requisition Management data screen 110 shown in Appendix II, the number 218848 has been entered in the Account Number field, and the notation "TEST NEW ONE" has been entered in the Requisition Number field.

The user can next enter desired items and quantities for the requisition. Each desired item may be identified by

entering its distributor catalog or part number, if known, in the field below the STOCK NBR label on the appropriate line in Requisition Item Table 46 shown on Requisition management data screen 110. In the sample Requisition Management data screen 110 shown in Appendix II, the part number 13246818F has been entered in the STOCK NBR field of Line 001. Once the user has entered such information at least partially describing a desired item on Requisition Management data screen 110, he or she may wish to initiate a search of catalog database 36 to find all the part numbers contained in catalog database 36 that match the part number entered or other information on Requisition Management screen 110. If so, the user enters the letter "S" (for "Select") on the line number of the item that he or she wishes to search in catalog database 36. The letter "S" has been entered to the left of line 001 on the sample Requisition Management data screen 110 shown in Appendix II. Any number of items, or no items, listed on Requisition Management data screen 110 may be marked with "S."

A user may not always have information relating to the catalog or part number for the particular items that are to be requisitioned using Fisher RIMS system 40. Or, the user may have relevant information about an item from a particular vendor but may wish to locate information about the same or a similar product available from other vendors. Or, the user may simply know the name of the item that he or she wishes to requisition. In any of these cases, the user alternatively or additionally could enter text at least partially describing the product to be requisitioned in the "DESC" field of Requisition Management data screen 110 (e.g., Appendix II). Then, the user would initiate the electronic sourcing system 5 of the present invention to search the vendor product catalogs contained in catalog database 36. Alternatively, the user could initiate search program 50 of electronic sourcing system 5 without having first entered information in RIMS system 40 about the product to be requisitioned.

Once the user has built or partially built Requisition Item Table 46 by filling the line numbers (entries) on Requisition Management data screen 110 and selecting those lines to be searched, he or she is now ready to initiate electronic sourcing system 5. Pressing the F11 function key, which is labelled "Catalog," from Requisition Management screen 110 accesses electronic sourcing system 5.

Referring now to FIG. 2, after the user presses the F11 key on Requisition Management data screen 110 of Fisher RIMS system 40, Fisher RIMS system 40 will pass program control via XCTL 74 to ESRC program 70. XCTL 74 is a protocol within CICS application 34 that directs the execution of a program, as would readily be understood by one of ordinary skill in the art. As control is passed from REQI program 44A to ESRC program 70, ESRC-Comm-AREA data structure 76 is passed. ESRC-Comm-AREA is a layout of storage area in local computer 20 created by REQI program 44A to pass data to ESRC program 70, as would readily be understood by one of ordinary skill in the art. ESRC program 70 will then LINK 82 to ESCP program 80 with ESCP-Comm-AREA 84. LINK 82 is a protocol within CICS application 32 that directs the execution of a program, as would readily be understood by one of ordinary skill in the art. Data at least partially describing one item desired to be requisitioned is passed to ESCP program 80 via LINK 82. Thus, if there are five items to be passed to ESCP program 80, there will be five LINKS 82 made. If no items are to be passed to ESCP program 80, only one LINK 82 is made to ESCP program 80. ESCP program 80 can return up to twenty items per LINK 82; in other words, for each item desired to

be requisitioned up to twenty desired catalog items contained in catalog database 36 may be sent to REQI program 44A and its associated Requisition Management data screen 110 of Fisher RIMS system 40. If a user chooses to terminate the sourcing process, ESRC program 70 would return to REQI program 44A and its associated Requisition Management data screen 110 without processing any of the records.

ESCP program 80 links with Shell 52 and TV/2 search program 50 via DDE LINK 90. Shell 52 and TV/2 search program 50 search in catalog database 36 for the item or items desired to be requisitioned that has or have been passed from ESRC program 70 to ESCP program 80. Catalog database 36 contains the following fields: vendor name, vendor number, vendor part (catalog) number, product description, list price, page number, quantity, unit, catalog text, and catalog images. Shell 52 and TV/2 search program 50 may, if desired, search the keyword field or any other field shown in Appendix VII. However, not all fields may appear on the monitor 22 of local computer 20, although they are stored in memory.

After the user has pressed the F11 key from Requisition Management data screen 110 and control has been passed from REQI program 44A to Shell 52 and TV/2 search program 50, monitor 22 of local computer 20 will show a footer bar representative of Shell 52 at all times that the user is in the TV/2 search program 50. The footer bar, which also includes appropriate icons, is used to make choices within Shell 52. A sample of the footer bar (without the icons) representing Shell 52 is shown at the base of Appendices III-VII. In the screens of Appendices III-VI, this footer bar is active to select functions. In the screen of Appendix VII, this footer bar is in the background and another footer bar is used to select functions.

If the user has marked an item on Requisition Management data screen 110 with the designation "S," the entered data at least partially describing that item will be sent to Shell 52 and TV/2 search program 50A in the manner described above. TV/2 search program 50 will search catalog database 36 for all items that match the search field sent over from REQI program 44A and Requisition Management data screen 110. When a search is performed in Shell 52 and search program 50, a Hit List 47 is produced, as indicated in FIG. 1C. The user would see on monitor 22 of local computer 20 a Hit List 47 screen representing limited data about all matching catalog items that were located in catalog database 36 as a result of the search. A sample Hit List 47 produced from a search initiated when the entry "OVENS" is received as the description or keyword by search program 50 from Requisition Item Table 46 is shown in Appendix III. Similar Hit Lists 47 are produced when various searches are performed from the Search Input screen shown in Appendix VII. When a Hit List 47 is depicted on monitor 22, the underlying catalog text and pictures (in either partial or complete form) are typically collected in a memory location for rapid viewing, printing or other use.

When multiple catalogs are present in catalog database 36, search program 50 contains a function associated with the catalog symbol of the footer bar and screen window (not shown) for selecting catalogs to be searched. For example, the following choices might be available:

1. Fisher General Catalog 93-94;
2. Fairmont Supplies Catalog;
3. NIST Standards Catalog; and
4. Promega Biological Research Products Catalog.

Fairmont and NIST catalogs list products not in the Fisher General Catalog, but many of the products listed in the

Promega catalog are also listed in the Fisher General Catalog (identified by corresponding Fisher catalog numbers). If searching for a molecular biology product, the user would select the Fisher and Promega catalogs. TV/2 search program 50 would then concatenate those two catalogs to perform a keyword, catalog number or other subject search and generate a Hit List of pages (panels) from both catalogs where the searched-for items were found. Similarly, the user might select the Fisher and NIST catalogs when searching for quality control standards or might select the Fisher and Fairmont catalogs when searching for supplies.

If the search is initiated from requisition/purchasing program 40, for example from the Requisition Management data screen 110 of the Fisher RIMS system, then the catalogs searched can be determined by the information provided. If, for example, Promega is indicated as the desired requisition item vendor, interface 60 would direct TV/2 search program 50 to search the Fisher and Fairmont catalogs. If no catalog delimiting information is entered for the item desired to be requisitioned, interface 60 would be set up to search only the Fisher catalog or, alternatively, to search all catalogs in catalog database 36.

Once Hit List 47 has been created by TV/2 search program 50, the user can view it and select particular ones of the located catalog items for Order List 48 that is being created in Shell 52, as shown in FIG. 1C. For example, a search for "Eco RI," a restriction enzyme, may have uncovered five entries in the Promega catalog (identified by Promega catalog numbers R6011, R6012, R6013, R6015 and R401) and five entries in the Fisher catalog (identified by Fisher catalog numbers PRR6011, PRR6012, PRR6013, PRR6015 and PRR4014). If the user selected PRR6012 from the Fisher catalog, Fisher catalog number PRR6012 would be added as an entry to the Items Selected screen, with VN00000001 (identifying the vendor as distributor Fisher) accompanying it in the Order List 48. If the user instead selected the item identified by catalog number R6012 from the Promega catalog, then Promega catalog number R6012 would be added as an entry to the Items Selected screen, with VN00005860 (identifying the vendor as Promega) accompanying it in the Order List. In either case, the information transmitted to REQI program 44A of Fisher RIMS system 40 would also include description, list price and other information taken from the catalog database from which the selection was made. When the resultant requisition is sourced, however (as described below), Distributor's mainframe host computer 10 would recognize the entry for the item from vendor Promega's catalog (R6012, 00005860) as corresponding to that same item available from Fisher's catalog (PRR6012, 00000001). The system thus would transmit back the Customer's contract price and availability for corresponding item PRR6012 as a type 03 (regular Distributor) product available from one of distributor's inventory locations. A purchase order then would be generated for this corresponding Distributor item as further described below.

By contrast, an item selected from the Fairmont catalog would be transferred to Fisher RIMS system 40 with the vendor number for Fairmont, and would be recognized during inventory sourcing as either a type 07 product (that Distributor orders from Fairmont) or as a type 05 item (that Customer orders from Fairmont as an Administrative Purchase). In either of these two cases, a purchase order would be generated for an item, corresponding to a desired catalog item, that is identified by the same Fairmont catalog number that was requisitioned.

After the desired item has been selected from the Hit List 47 by double clicking on that item TV/2 search program 50

11

can be used to bring up for viewing on monitor 22, or printing on printer 26, images and text from the catalog page on which the item selected is located. For example, as shown in Appendix III, page 1106 of the Fisher catalog has been selected. If the user double clicks on highlighted page 1106, the text shown in Appendix IV (and related images, not shown) would appear on monitor 22. On the sample screen shown in Appendix IV, the item that appears on page 1106 of the Fisher catalog relates to Fisher Isotemp 800 Series Programmable Ovens. Conventional scroll bars appearing on the screen (not shown in Appendix IV) enable the user to scroll through additional catalog information (text and/or images) not yet displayed on the screen. An example of such additional textual information is depicted on the screen shown in Appendix V.

On the screen of Appendix V, the vendor distributor's catalog number ("Cat. No.") 13-246-818F is highlighted. The catalog number of an item normally appears in blue in a screen such as Appendix V. This blue lettering is used for catalog numbers, trademarks, footnotes and other entries for which database 36 contains additional information or cross-references (called hyperlinks). When a search is conducted and the catalog segments of the resultant hit list are reviewed, the text corresponding to the search parameter is highlighted in red. Thus, in Appendix V, catalog number 13-246-818F (identified in the search) appears in red, while catalog number 13-246-838F and the trademark Isotemp each appear in blue. A word, vendor part number or catalog number located by the search will appear red, even if that word or number did not have an associated hyperlink (and thus is not normally blue).

When in search program 50, particular items selected can be added to an Order List 48 pending in Shell 52 and search program 50. When the Ordering portion of catalog text is viewed (as in Appendix V), particular items can be selected so as to be added to the Order List 48 by double clicking on the highlighted catalog number (even if a different field was also highlighted as a result of a search of catalog database 36). The item is then added to an Order List 48 that is created in Shell 52 via a hypertext link. The items that are sent to the Order List 48 are collected and shown on the Items Selected screen of Shell 52. An example of an Items Selected screen of Shell 52 is shown in Appendix VI. The Items Selected screen depicts certain fields of Order List 48 that can be viewed and edited within search program 50. For example, Shell 52 permits the user via a pop-up window (not shown) to select units, e.g. pack or case, and quantity to be ordered, e.g. two packs. Alternatively, the data in these fields can default to one of the smallest unit and the units can be changed when the order is reviewed in REQI program 44A. Additional fields on the same items are also present in memory at this stage. Upon clicking on "Order" when the Items Selected screen (Appendix VI) is viewed, many or all of these fields on the items in the Order List are transmitted back to REQI program 44A (via the programs of interface 60 shown in FIG. 2) to be added to the pending Requisition Item Table 46. The sample Items Selected screen shown in Appendix VI includes the Isotemp Oven with catalog number 1324818F that was located as a result of the search for all items in catalog database 36 that match the part number 13246818F that was entered in the STOCK NBR field of REQI program 44A and its associated Requisition Management data screen 110 of Fisher RIMS system 40.

The following fields are transferred to Order List 48 created in TV/2 search program 50: Vendor name, vendor number, vendor part (catalog) number, product description, list price, page number, quantity, unit and catalog text.

12

However, not all of these fields are viewed on the Items Selected screen.

If more than one item on Requisition Management data screen 110 had been marked with an "S," the process described above is repeated.

If the user desires to do additional searching in catalog database 36 that is not connected to catalog or other items that have been listed on Requisition Management data screen 110 of Fisher RIMS system 40, he or she can click the box on footer bar of Shell 52 that is labelled "Search." Then, a Search screen comes up on monitor 22 of local computer 20. An exemplary Search screen is shown in Appendix VII. In this screen, the usual footer bar is visible in the background, but is not active.

Using the Search screen, a user can search catalog database 36 by page, text description, part number (where the user has the further option to search by Fisher part number, for example if Fisher is to be the desired vendor), Vendor part number, vendor name (for vendors other than Fisher), or bulletin. Stock numbers specific to the customer can also be present in catalog database 36 and searched using the screen of Appendix VII. "Bulletin" refers to an additional vendor publication with detailed product information that may not be included in a vendor catalog. Searching for information contained in bulletins may be done by bulletin number, but only if bulletins have been made a part of catalog database 36. For purposes of this disclosure, bulletins when included in a catalog database are considered a type of catalog.

After the user has entered the field to be searched on the Search Screen, the user clicks on the "SEARCH" box near the bottom of the Search Screen. A Hit List 47 indicating all items from catalog database 36 that match the search field that was entered on the Search Screen then is generated. Then, in a manner similar to that described previously, the user can scroll through the Hit List 47 and double click on the catalog page or panel desired. The user may then also view the detailed information located on the catalog page that was selected from the Hit List 47. During the search, the user may also add additional items to the Order List 48 being built in Shell 52 if desired, whether those additional items had been selected from the Hit List 47 or not.

The Order List that the user has built in Shell 52 is maintained on the Items Selected screen, shown in Appendix VI. From the Items Selected screen, the user can cancel the order by clicking on the "Cancel" box at the bottom of the screen, delete an item from the Order List 48 by moving the pointer bar to the item to be deleted and then clicking on the "Delete" box at the bottom of the screen, or delete all items by clicking on the "Delete All" box. The user can also view catalog text and images for a particular item by clicking on the "Description" box.

Once the user has completely built the Order List 48 within Shell 52 and TV/2 search program 50, he or she can transmit it to Fisher RIMS system 40. This is accomplished by clicking on the "Order" box at the bottom of the Items Selected screen to communicate the completed Order List 48 to Fisher RIMS system 40.

The user may have selected no items, one item or several items from the catalogs contained in catalog database 36 by using TV/2 search program 50. If no items have been selected, the original items that were entered on Requisition Item Table 46 of Requisition Management data screen 110 will remain on that screen and will continue to be processed by Fisher RIMS system 40. If one or several desired catalog items were selected in TV/2 search program 50, the first item selected will replace the original item on Requisition Item Table 46 of Requisition Management data screen 110. Addi-

tional items that were selected from the search that was performed in TV/2 search program 50 will be added to Requisition Item Table 46 of Requisition Management data screen 110.

Interface programs ESCP 80 and ESRC 70 (FIG. 2) are used to send data to REQI program 44A (FIG. 1A) and its associated Requisition Management data screen 110 (FIG. 2) about the items that were selected from the search performed by TV/2 search program 50. To the user, it appears that all the items selected from the search are sent over to Fisher RIMS system 40 at the same time. However, ESCP program 80 receives multiple items from TV/2 search program 50, and then sends one item at a time to ESRC program 70. ESRC program 70 then waits until all items have been passed to it before sending data about the items to REQI program 44A and its associated Requisition Management screen 110 of Fisher RIMS system 40. The information transmitted to Requisition Management screen 110 from the Order List built in TV/2 search program 50 and sent through ESCP program 80 and ESRC program 70 includes vendor name, vendor number, vendor part (catalog) number, product description, list price, page number, quantity, unit and catalog text. However, not all of the above-listed fields may be displayed on screen at all times. ESRC program 70 passes control back to Fisher RIMS system 40 via XCTL 78. The requisition number, customer identification and release number (or other data identifying the requisition) will be passed in MENU-Comm-AREA 56 to confirm that the returned data are associated with the proper requisition. MENU-Comm-AREA 56 is a layout of storage area within local computer 20, as one of ordinary skill in the art would readily understand.

As previously indicated, multiple LINKS 82 may have been created between program ESRC 70 and program ESCP 80 if multiple lines were selected (with the "S" symbol) in Requisition Management data screen 110. After completing the first search, and any additional searches initiated with the footer bar, an order list is created and returned to Requisition Item Data Table 46 associated with Requisition Management data screen 110. At this point, the next item is sent from a LINK 82 through program ESCP 80 and DDE LINK 90 to the TV/2 program 50, and a hit list resulting from the corresponding search is displayed on monitor 22. The process of searching, displaying, selecting and ordering is repeated until all of items stored by LINKS 82 have been sent to TV/2 program 50 and searched. At the end of each of these searches, an order list may be created and returned to Requisition Item Data Table 46 or cancelled. Once the last item is completed, ESRC program 70 passes control via XCTL 78, and a Requisition Management screen 110 is displayed, reflecting all of the additions and changes that have been made to the Requisition Item Data Table 46 associated with that requisition.

A limit is normally placed on the number of items of an order that may be returned to the Requisition Item Data Table 46. For example, if the maximum size in Requisition Item Data Table 46 is set at 200 lines, one could create a limit on the size of each order list at 20, 50, 100 or even 200. A corresponding limit can be placed on the number of LINKS 82 that can be established concurrently from the same requisition. Setting a limit of five LINKS 82 and forty items per order list would be one way of avoiding situations in which a Requisition Item Data Table 46 reaches its limit (e.g., 200 lines) before all of the searches (five) have been completed and order lists (five of forty items each) have been returned.

At this point in the use of Fisher RIMS system 40, as many entries (lines) of Requisition Management data screen

110 have been built up (some through use of electronic sourcing system 5) as are necessary to complete the requisition. A sample of such a Requisition Management data screen 110, in which four lines have been entered identifying desired items to be requisitioned (including catalog items located as a result of a catalogs search), is shown in Appendix VIII. The next step is that of inventory sourcing using RIMS inventory sourcing program or programs 44B in Fisher RIMS system 40, as shown in FIG. 3. Inventory sourcing is the process of determining what inventory will be used to fill the requisition. Pricing is also performed in this step when it is called for. Inventory sourcing in Fisher RIMS system 40 is performed on both local computer 20 and host computer 10.

Within Fisher RIMS system 40, a Requisition Item Table 46, as shown in Appendix VIII (similar to that shown in Appendix II, but including more items), can be inventory sourced by pressing the key F6 from REQI program 44A represented by Requisition Management data screen 110 shown in Appendix VIII (and in Appendix II). Since inventory records on JIT items (type 01 and 06) are maintained in inventory database 42B, lines 002 and 004 in Appendix VIII show the availability of these items in inventory (49 items available for line 002, and 0 items available for line 004). After the F6 key has been pressed, host computer 10 searches its host pricing and inventory databases for availability of the various items listed on Requisition Management data screen 110 in different inventory locations (e.g., different warehouses) as described in further detail, below.

After such inventory sourcing, and assuming that no errors occurred during sourcing (as indicated by decision step 116 in FIG. 3), the contract price, source (inventory) location and available quantity or other fields are communicated back to computer 20 by host computer 10, and entered and displayed in the Requisition Management Screen. This can best be seen by comparing lines 001 and 003 of Appendix VIII to Appendix IX, especially as to "QTY AVAIL" (quantity available), "LOC" (inventory location) and price. As Appendix IX indicates, an inventory-sourced Requisition Item Table 46 typically contains the same items, but with more completed fields (including price, product type and inventory location). Moreover, as discussed above, an entry in an inventory-sourced Requisition Management screen may indicate for a requisitioned item a vendor and vendor catalog number that has been changed, from what was obtained from a catalog search, to a corresponding vendor and vendor catalog number for that item from another source (e.g., Fisher—which has its own catalog number for that manufacturer's item that Fisher distributes).

For example, as shown in Appendix IX, product type "01" for the item on line 002 indicates that the requested requisition item is available as Distributor-owned inventory in the JIT inventory that the vendor/distributor maintains near local computer 20, either for the particular Customer or for a group of customers. Product type "06" for the item on line 004 indicates that this item is available for the requisitioner employed by the Customer from inventory owned by Customer's purchasing department but managed by local computer 20. Product type "03" for the items on lines 001 and 003 indicates that these are regular Distributor items that the communication between Distributor's host computer 10 and local computer 20 determined were available in sufficient quantity at one or another of Distributor's general warehouses designated "DEL" and "EDC" in the location ("LOC") field. Product type "05" (not shown in Appendix IX) indicates that a requisitioned item is to be purchased by Customer directly from an outside supplier, using an Admin-

15

istrative Purchase Order that local computer 20 creates and prints (or transmits) for Customer.

The inventory sourcing process described above also determines the net prices shown in Appendix IX for each item. Type 01 and type 03 items are priced by Distributor's host computer 10 searching host databases 11, which contain various formulae and tables of Distributor's pricing agreement with the Customer. Host computer 10 also prices any type 04 or type 07 item, if present. These prices were transmitted to local computer 20 along with the location and availability information for the type 01 items. Prices for type 05 and 06 items are maintained in the local computer's 20 own databases 42B and 42C.

From Requisition Maintenance data screen 120, the CSR can accept all lines of the requisition—if all lines show the status "S" for sourced in the "STAT" field of Requisition Maintenance data screen 120—by pressing the F6 function key. If item errors are found at step 116 in the data transmitted back to local computer 20 from host computer 10 during the sourcing process, then those particular items for which error was found will be returned and displayed by local computer 20 in Requisition Management data screen 110.

Once a requisition has been inventory sourced and accepted by the CSR, it can be converted to one or more purchase orders, as represented by step 114 in FIG. 3. For example, the requisition represented by the Requisition Item Table 46 of Appendix IX, if accepted without further revision by pressing function key F6 ("ACCEPT"), would result in the generation of the following three purchase orders:

- A. Line 002 would be ordered from on-site distributor-owned inventory;
- B. Line 004 would be ordered from on-site customer-owned inventory (a transfer internal to the customer); and
- C. Lines 001 and 003 would be ordered, respectively, from Distributor's "DEL and "EDC" warehouses.

Of these three purchase orders, Orders A (type "01") and C (type "03") are shared between host computer 10 and local computer 20 (as shown in FIG. 3). Upon execution of Order A, the inventory records on both computers for Distributor-owned JIT inventory are adjusted synchronously. A purchase order is generated by host computer 10 immediately thereafter. Order B (type "06") is executed and stored only on local computer 20. Upon execution of Order B, the inventory record on local computer 20 is adjusted (the host computer contains no records on Customer-owned JIT inventory or on items ordered by Administrative Purchases). For Administrative Purchases (type 05 items), a purchase order is printed, and mailed or faxed, locally by computer 20 as indicated at step 118 in FIG. 3, or via host computer 10 via EDI (if EDI was selected in the Header of Appendix I and an EDI transfer arrangement existed with vendor).

It is an important feature of the present invention that a requisition may be filled by searching and selecting from a catalog database of items, inventory sourced, and the resulting requisition then divided into one or more purchase orders. This contrasts with known prior art CD-ROM catalog systems in which only a single purchase order to a single supplier is built without reference to inventory records, and in which the information used to create the purchase order is limited to that contained in the product catalog of a single vendor.

Electronic sourcing system 5 also contains the capability to log messages returned from inventory sourcing program or programs 44B of Fisher RIMS system 40. Messages will be logged for any of the following reasons: (1) part number

16

changes for line sent to ESCP program 80; (2) list price from inventory sourcing program 44B differs from list price returned from ESCP program 80; (3) vendor name from inventory sourcing program 44B differs from vendor name returned from ESCP program 80; (4) on a "master or blanket" order, in which local computer 20 tracks the amount of purchases against a blanket or cumulative sum available and/or in which there is limited access to products or limited access to certain users, the part has already been entered on another line; and (5) the maximum number of line items has been reached.

Referring again to FIG. 2, a user is able to view the messages returned by pressing the ALT F11 function keys in REQUI program 44A and its associated Requisition Management screen 110 in Fisher RIMS system 40. After the ALT F11 keys have been pressed, REQUI program 44A will link to ESMV program 112 via XCTL link 111 for displaying the message log created. ESMV program 112 is a function of Fisher RIMS system 40. ESMV program 112 allows the user to page through the messages created and then to return to Requisition Management screen 110. A sample ESMV message screen associated with ESMV program 112 is shown in Appendix X.

The first two messages of the message screen of Appendix X indicate that a part number for line 001, identified as part number 53610, was successfully added in substitution for a prior part originally entered as part number S100-06 (from the Fisher Scientific catalog). These messages were generated because the originally entered part (S100-06) did not exist in the Fisher catalog, but its corresponding part number S100-06 (that was located by another search in another catalog) did exist in that other catalog. The next message indicates that the vendor for part number 53610 was changed in line 001 from "VN00000001"—meaning that the originally requested vendor (Fisher) was changed. The next two messages indicate that two other part numbers (53620 and 53650) were successfully added as lines 002 and 003.

In the previous description, an exemplary embodiment has been described in which a Distributor CSR operates Fisher RIMS requisition/purchasing system 40 and IBM TV/2 search program 50 as part of a Just-In-Time activity for a particular customer, Customer. Electronic sourcing system 5 of the present invention may also be used, however, in other requisition and purchasing environments.

In some embodiments, a Customer end user or a Customer purchasing employee operating REQUI program 44A of Fisher RIMS system 40 may also operate TV/2 search program 50. Operating either from a terminal connected to local computer 20, or from a separate local computer networked with the CSR's local computer 20, such a Customer end user can select requisitioned items for inclusion in Requisition Item Table 46 by keystrokes viewing that screen and by searches in TV/2 search program 50 which are transmitted to the Requisition Item Table 46 via interface 60, as described above. Depending upon his or her authorization level and access code to Fisher RIMS system 40, the Customer purchasing employee may be able to source the final requisition and/or accept the sourced requisition, as shown in Appendix IX. If, however, the sourced requisition was split into more purchase orders than the Customer purchasing employee might prefer, the intervention of the Distributor CSR could be invoked to revise and re-source the requisition (causing, for example, certain items originally sourced as type 01 products to be sourced for this order as corresponding type 03 products from a common Distributor warehouse with other type 03 products on the requisition). The Customer end user may have authority only

to build the Requisition Item Table, but then calls the Distributor CSR or Customer purchasing employee to source and accept the requisition.

As shown in FIG. 1B, the present invention also has application to Distributor's regional customer service locations where a large number of CSRs may be placing orders directly on Distributor's host computer 210 for thousands of different customers who call in. In that environment, search program 250, which preferably comprises TV/2 search program 250, and catalog databases 236 are stored on file server 200. In this environment, file server 200 is a large personal computer, a work station or a mini-computer such as an IBM AS/400. Alternatively, the server 200 and a minicomputer (such as an IBM AS/400) can be independently connected to each local computer 200. Each CSR has a local personal computer 220 having a monitor 222, a keyboard 224 and a printer 226. Local computer 220 is provided with programs including requisition/purchasing program 240, Shell program 252 and a graphic user interface 254 (preferably EASEL Workbench program 254 for OS/2) for listing items. One or more of these may be copied from server 220 when needed. Work-in-progress requisitions 260 are established for each customer and are attached to graphic user interface 254. Server 200 maintains complete requisitions 242, in a manner similar to the manner in which local computer 20 maintains requisition databases 42 in the embodiment shown in FIG. 1A.

Normally, in such an environment, the CSR creates Order lists for customers by entering Distributor catalog numbers into graphic user interface 254 and connecting to the Distributor mainframe 210 for price and availability. For this purpose, each local computer is connected to host computer 210 via a phone/dataline and either a gateway or a mini-computer acting as a local host. When a customer asks for products by manufacturer part number or a competitor's catalog number, the CSR has access to cross-reference files, as earlier described, either maintained on the local host or maintained on the Distributor host computer 210.

Appropriate Distributor catalogs and manufacturer catalogs then are consulted, using TV-2 search program 250 and proper selection of Distributor catalogs and of catalogs and bulletins from manufacturers whose products Distributor regularly sells. Catalogs and bulletins are contained in catalog database 236. The resultant lists of products are then transferred by Shell program 252 to a work-in-progress requisition 260, and then entered from graphical user interface 254 directly onto Distributor's mainframe computer 210 as orders from the applicable customer to Distributor. The CSR, knowing which items are available from which Distributor warehouse and direct-shipping supplier, then may divide the customer's requested items into multiple orders, so as to assure that each order is completely filled by a single shipment. In this regional environment, file server 200 or the minicomputer acting as local host can maintain files of completed requisitions 242 which can be subsequently used for generating reports for customers in the region. Reports can be generated either from such local data or from data periodically downloaded to the local host from Distributor's host computer 210.

Another environment where the present invention can be used is in Distributor's purchasing department. The item lists created in that environment can include lists of items Distributor does not regularly stock or purchase, but for which particular customers indicate a requirement to buy. The file server 200 in that environment contains TV-2 search program 250, EASEL graphical user interface 254 and multiple catalog databases 236 containing catalogs similar to

the Fairmont and NIST catalogs described above for the embodiment of FIG. 1A. The Distributor purchasing employee can receive by phone or via Distributor's host computer 210 requests for items not shown on Distributor's host databases either as regular products (type 03) or third party items purchased for particular customers on a regular basis (type 07 items). Transmitting certain such requirements to the applicable Distributor purchasing employee can be a function of the inventory sourcing routines of host computer, or may be directed by the Distributor CSR interfacing with the customer.

The Distributor purchasing employee can search appropriate catalogs using TV-2 search program 250, and can transfer the "Items Selected" to a product list in EASEL interface 254. The resultant list might display, for example, supplier part number, supplier, list price, product and catalog page, with access to other fields such as complete description (up to 500 characters). The Distributor purchasing employee can then either forward the information to the CSR, customer end user or customer purchasing employee who requested the item (to confirm that the requirement is being met) or contact the supplier to confirm pricing and availability. Once responses from either or both have been obtained, the Distributor purchasing employee can use the item list in EASEL interface 254 to create one or more of the following purchase orders:

1. an order from the customer to the supplier (an Administrative Purchase);
2. an order from the customer to Distributor (for a type 07 product); and
3. an order from the Distributor to the supplier (usually providing for direct shipment from the supplier to the customer or to a JIT site maintained by Distributor for the customer).

From the foregoing description, it should be apparent that the network arrangements of FIG. 1B can be used to apply the present invention in a variety of contexts. The context will dictate which catalog databases 236 are provided on file server 200: in the regional CSR environment, Distributor's catalogs can be present with a variety of catalogs and bulletins from manufacturers that Distributor regularly represents and a limited selection of outside suppliers; and in the Distributor purchasing environment, the number of outside supplier catalogs will be increased. The number of client (local) computers 220 and the number and size of catalog databases 236 will help dictate what size file server 200 is required. The operating environment (regional CSR site, on-site CSR, on-site CSR networked with Customer end users and with purchaser personnel or Distributor purchasing site) will also affect the catalog databases 236 included, file server 200 size and requisition/purchasing program 240 used. In some situations (e.g., purchasing) each client computer has an independent copy of requisition/purchasing program 240; in others (e.g., on-site CSR) a single copy of the requisition/purchasing program 240 is maintained with associated local databases on the server 200. Where the requisition/purchasing program 240 and local databases are maintained on file server 200, the local database is updated after each use for the benefit of subsequent users. For example, in an environment using Fisher RIMS for requisition/purchasing program 240, if a NIST standard is selected using TV-2 search program 250 and ordered using Fisher RIMS 240 (as either a type 07 purchase from Distributor or a type 05 administrative purchase from NIST), that item is available in the applicable database for subsequent requisitions. For example, a NIST standard ordered as a type 05 item will be stored in the local database

on file server 200, with NIST as the vendor for subsequent administrative purchases by Customer. A NIST standard ordered from Distributor as a type 07 item will be stored in Distributor's host databases as a type 07 available to Distributor from NIST. The local databases on file server 200 will also contain records of all items requisitioned and ordered, useful to transfer files to a Customer's computer (e.g., of purchase orders placed by that Customer in a day) or to generate reports for a Customer (e.g., or requisitions placed by each Customer department and/or budget number in a week).

Thus it is seen that an electronic sourcing system including means for linking a requisition/purchasing system and a means for searching large volumes of information has been described. Persons skilled in the art will appreciate that the present invention can be practiced by other than the described embodiments, which are presented for the purposes of illustration and not of limitation, and the present invention is limited only by the claims which follow.

APPENDIX I

FISHER SCIENTIFIC RIMS
REQUISITION HEADER

DATE: 08/05/94
TIME: 07:04:57
ACCT-NBR: NAME:
ADDRESS:
COMPANY: :
REQ NBR: :
RELEASE: ORDER TYPE: R ORDER
HOLD/REL: I RUSH CODE: 9
CALLER: FREIGHT OVERRIDE: N TAX OVERRIDE:
EDI PO TO HOST: N POA 855
ATTN: PRT ACK: Y NBR OF COPIES: 1
ACK DELV CODE: F PRINT & DELIVER
BILL TO: REQ DELV CODE: W WALK IN
SERVICE CHARGE: 0.00
VENDOR: CREATED: 08-04-1994 STATUS: R
RESPONSE: KEY(S):
+F2: ADD F3: EXIT F4: UPDATE F5: REFRESH F6: ITEM F9: VAR F10: SRCE F11: CHGPO F12: DEL
13V123

APPENDIX II

*** REQUISITION MANAGEMENT SCREEN ***

ACCT NBR: 218848 002 REQ NBR: TEST NEW ONE
COMP: 1 REL NBR:
S LINE STOCK NBR QTY UM PT STKRM XREF SPI UNIT PRICE EXT PRICE
001 13246818F 0 CS 03 0.00 0.00
DESC: QTY AVAIL: 0 LOC: FSHR WHSE: BLW
002 QTY AVAIL: LOC: WHSE:
003 QTY AVAIL: LOC: WHSE:
004 QTY AVAIL: LOC: WHSE:
005 QTY AVAIL: LOC: WHSE:
DESC: QTY AVAIL: LOC: WHSE:
RESPONSE: KEY(S):
ALL ITEMS DISPLAYED
F3: EXIT F6: SOURCE F7: BKWD F8: FWD F9: NEW F10: NONCAT F11: CATALOG F12: CNCL

APPENDIX III

ovens
General
(1106)Fisher Isotemp 800 Series Programmable Ovens
(1107)Isotemp 700 Series Deluxe Lab Ovens
(1108)Isotemp 600 Standard Lab Ovens
(1109)Fisher Isotemp 500 Series Economy Lab Ovens
(1110)Gravity Convection Ovens
(1111)Utility ovens
(1112)Mechanical Convection Ovens with Electronic Temperature
(1113)General-Purpose Ovens
(1114)Heavy Duty Deluxe Ovens

-continued

(1116) Large Capacity Model 2882A
 (1117) Standard Capacity Model 281A
 (1118) Fisher Models 280 and 285 Vacuum Ovens
 (1119) NAPCO Vacuum ovens

Help Catalogs Search Order List Minimize Clear Prev Next Exit

APPENDIX IV

(FSC1106) Fisher Isotemp 800 Series Programmable Ovens

Fisher Isotemp 800 Series Programmable Ovens

Three linear heat-up and cool-down stages

Talking control panel

Keypad and lighted graphics

30° to 325° C. range

RS-422 serial communications capability

The latest technology at your fingertips. Accurate, easy-to-use controls allow you to program up to 3 heat-up stages and 3 cool-down stages linearly to provide the most appropriate conditions for your samples. Using the large keyboard, you can choose the heat-up or cool-down rate, the temperature you want for each stage, and the length of time you want the oven to hold each temperature. And, for projects requiring repeatability, you can duplicate the settings at any time.

Help Catalogs Search Order List Minimize Clear Prev Next Exit

APPENDIX V

(FSC1106) Fisher Isotemp 809 Series Programmable Ovens

Model	818F	838F
Inside D x W x H	16 x 12 x 16 (41 x 30 x 41 cm)	18 x 18 x 20 (46 x 46 x 51 cm)
Shp. Wt.	156 lb. (71 kg)	195 lb. (88 kg)
Electrical Requirements	230 V 50/60 Hz 11.3 Amps	230 V 50/60 Hz 19 Amps
Cat. No.	13-246-818F	13-246-838F
Each	3495.00	3995.00

Extra Shelves for 800 Series Ovens

No-tip design. Move to any position in seconds. Full Depth Shelves: Chrome-Plated Steel

Help Catalogs Search Order List Minimize Clear Prev Next Exit

APPENDIX VI

ITEMS SELECTED

Part Number	Description	List Price
13246818F	ISOTEMP OVEN MDL818F 230 V	3495.00

Help Cancel Delete Delete All Order Description

APPENDIX VII

SEARCH

Page:

Search For:

Part Number: ☐ Fisher ☐ Vendor ☐ Customer

Vendor Name:

Bulletin:

HELP SEARCH CANCEL CLEAR USER DATA EXTENDED

Help Catalogs Search Order List Minimize Clear Prev Next Exit

APPENDIX VIII

RICREQ11

FISHER SCIENTIFIC RIMS
REQUISITION MANAGEMENT SCREEN

DATE: 07-29-94

TIME: 14:54:22

ACCT NBR: 363690 006 REQ NBR: PO NBR 001
 COMP: 1 REL NBR:

0 LINE	STOCK NBR	QTY	UM	PT	STKRM	XREF	SPI	UNIT PRICE	EXT PRICE
001	A191	1	EA	03				0.00	0.00
DESC:					QTY AVAIL:	0	LOC:	F5HR WHSE:	EDC
002	02540K	1	PK	01				0.00	0.00
DESC:					QTY AVAIL:	49	LOC:	WHSE:	JIT
003	13246818F	1	EA	03				0.00	0.00
DESC:					QTY AVAIL:	0	LOC:	F5HR	WHSE: EDC
004	A181-06	1	EA	06				100.00	100.00
DESC:	ACETONE				QTY AVAIL:	0	LOC:	WHSE:	JIT
JIT BACKORDER WILL OCCUR									
005									
DESC:					QTY AVAIL:	0	LOC:	WHSE:	
RESPONSE:	KEYS(S):								
1 ITEM(S) PROCESSED									

-continued

+F3: EXIT F6: SOURCE F7: BKWD: F8: FWD F9: NEW ITM F10: NONCAT
F11: CATALOG F12: CNCL

1B V123

APPENDIX IX

RICPOMP1 FISHER SCIENTIFIC RIMS DATE: 08-03-94
REQUISITION MANAGEMENT SCREEN TIME: 07:44:13

COMP ID: 001 REQ-NBR: PO NBR 001
ACCT NBR: 363690 006 REL-NBR:

ORDER NBR: PICKLIST REVIEWED:
SERVICE: 0.00 ORDER: 0.00 FREIGHT:

CARRIER:

O LINE	PART	QTY	UOM	PRD	UNIT PRICE	SERVICE	EXT PRICE	LOC	STAT
001	A181	1	EA	03	35.30	0.00	35.30	DEL	S
	ACETONE CERTIFIED ACS 1L QTY AVAIL: 1 QTY REC: 0								
002	02540K	1	PK	01	32.70	0.00	32.70	JIT	S
	BEAKER GRIFFIN 250 ML 12/9 QTY AVAIL: 49 QTY REC: 0								
003	13246818F	1	EA	03	3495.00	0.00	3495.00	EDC	S
	PROGRAMMABLE OVEN QTY AVAIL: 0 QTY REC: 0								
004	A181-06	1	EA	06	100.00	0.00	100.00	JIT	S
	ACETONE QTY AVAIL: 0 QTY REC: 0								

RESPONSE: KEY(S):
+F3: EXIT F6: ACCEPT F7: BKWD F8: FWD F9: PRINT ACK F11: M/B ERRORS F12 DELETE

1B V123

APPENDIX X

*** REQUISITION MANAGEMENT SCREEN ***

ACCT NBR: 218848 002 REQ NBR: TEST NEW ONE
COMP: 001 REL NBR:

ELECTRONIC SOURCING MESSAGES

LINE NUMBER	PART NUMBER
001	53610
PART ADDED SUCCESSFULLY	
001	53610
REPLACEMENT WAS MADE FOR PRIOR PART: S100-06	
001	53610
VENDOR CHANGED FROM: VN00000001	
002	53620
PART ADDED SUCCESSFULLY	
003	53650
PART ADDED SUCCESSFULLY	

F6: RETURN F7: BACKWARD F8: FORWARD

We claim:

1. An electronic sourcing system comprising:
 - a collection of catalogs of items stored in an electronic format;
 - a first set of pre-determined criteria associated with said collection of catalogs;
 - a second set of pre-determined criteria associated with items from each of said catalogs;
 - a catalog selection protocol, said catalog selection protocol relying on said first set of pre-determined criteria to select less than said entire collection of catalogs, and including matching a vendor identification code with a subset of said collection of catalogs, wherein said subset of catalogs includes both a vendor catalog from a predetermined vendor and a second catalog from a predetermined third party that is one of a manufacturer and a competing vendor, said predetermined third party selling items corresponding to items in said vendor catalog; and
 - a search program, said search program relying on said second set of criteria to select specific items from said catalogs determined from said catalog selection protocol.
2. An electronic sourcing system as recited in claim 1, wherein catalogs comprising said collection of catalogs are stored in separate databases.
3. An electronic sourcing system as recited in claim 1, wherein said catalogs comprising said collection of catalogs are stored in a single database.
4. An electronic sourcing system as recited in claim 1, wherein said predetermined third party makes items in said vendor catalog.
5. An electronic sourcing system as recited in claim 1, further including a cross reference table linking a vendor item catalog number from said vendor catalog with an item catalog number from said predetermined third party.
6. An electronic sourcing system as recited in claim 1, wherein said second set of predetermined criteria includes at least one of a catalog number and item textual information.
7. An electronic sourcing system as recited in claim 1, wherein said catalog selection protocol includes providing an electronic listing of available catalogs from said collection of catalogs.
8. An electronic sourcing system as recited in claim 7, wherein said electronic listing of available catalogs is less than said collection of catalogs.
9. An electronic sourcing system comprising:
 - a collection of catalogs of items stored in an electronic format;
 - a first identification code associated with a first item in a first catalog;
 - a second identification code associated with a second item in a second catalog, said first item and said second item

25

being generally equivalent, and wherein a selection of one identification code from one of said first and second catalogs provides the other identification code from the other of said catalogs.

10. An electronic sourcing system as recited in claim 9, wherein said first identification code is identical to said second identification code.

11. An electronic sourcing system as recited in claim 9, wherein at least one of said first and second catalogs includes said first and second identification codes.

12. An electronic sourcing system as recited in claim 9, wherein said selection includes a comparison of said one of said first and second identification codes with a cross-reference table listing both of said identification codes as being generally equivalent.

13. An electronic sourcing system as recited in claim 9, wherein a user selects one of said first and second identification codes, lacks access to said catalog corresponding to said selected identification code, but is given access to the other said catalog corresponding to said non-selected identification code.

14. An electronic sourcing system as recited in claim 9, wherein a user selects one of said first and second identification codes, and has access to both said first and second catalogs.

15. An electronic sourcing system as recited in claim 9, wherein said first and second identification codes correspond to a part number.

16. An electronic sourcing system comprising:
at least two product catalogs containing data relating to items such that an item in a first catalog is generally equivalent with an item in a second catalog; and
converting means for converting data relating to said item from said first catalog to data relating to said item from said second catalog.

17. An electronic sourcing system as recited in claim 16, wherein at least one catalog database contains said data from each of said catalogs, and said converting means includes a non-catalog database containing a cross-reference table such that use of a reference code corresponding to an entry in said cross-reference table links said item from said first catalog to data relating to said item from said second catalog.

18. An electronic sourcing system as recited in claim 16, wherein one or more catalog databases contain said data from each of said catalogs, and said converting means including one or more catalog databases including an identical reference code corresponding to said data from said first catalog and said data from said second catalog.

19. An electronic sourcing system as recited in claim 16, wherein said first catalog may be searched separately from said second catalog.

20. An electronic sourcing system as recited in claim 19, wherein a user lacks access to said first catalog and has access to said second catalog, such that a request for an item in said first catalog provides said data from said second catalog.

21. An electronic sourcing system comprising:
a requisition module including data fields, user-generated criteria entered into at least one of said data fields to generate at least partial criteria corresponding to a desired item;
a catalog collection searching module, said searching module including a collection of catalogs of items stored in an electronic format, a catalog selection criteria used to select less than said entire collection, said searching module being used to generate additional search-module criteria for said data fields of said requisition module;

26

a multiple purchase order generation module, said purchase order generation module creating multiple purchase orders from a single requisition created with said user-generated criteria and said search-module criteria; wherein each of at least two catalogs include a generally equivalent item from a different source, said requisition module working in combination with said catalog searching module to determine multiple sources for said item;

wherein said multiple sources is limited by said catalog searching module providing a match according to said user-generated criteria, said search-module criteria and a determination system that located items are generally equivalent; and

wherein said determination system includes a cross reference table matching an identification code from a first located item with a second identification code from a second located item.

22. An electronic sourcing system as recited in claim 21, wherein said determination system includes an identical identification code for each of said located items.

23. An electronic sourcing system, as recited in claim 21, wherein said requisition module generates a preferred requisition based on at least one of product availability and user preferences in accordance with a determination of multiple sources for a desired item.

24. An electronic sourcing system as recited in claim 21, wherein less than said catalog selection criteria is determined by at least one of said user-generated criteria or user characteristics.

25. An electronic sourcing system as recited in claim 24, wherein said user characteristics include a listing of catalogs from which a user is allowed to purchase.

26. An electronic sourcing module as recited in claim 21, wherein said requisition module uses at least one pre-determined rule to select which of multiple sources to use for said desired item.

27. An electronic sourcing system as recited in claim 26, wherein said pre-determined rule relies on item availability.

28. An electronic sourcing system as recited in claim 26, wherein said pre-determined rule relies on a hierarchy of preferred sources.

29. An electronic sourcing system comprising:
a collection of catalogs of items stored in an electronic format;

a first set of pre-determined criteria associated with said collection of catalogs;

a second set of pre-determined criteria associated with items from each of said catalogs;

a catalog selection protocol, said catalog selection protocol relying on said first set of pre-determined criteria to select less than said entire collection of catalogs, and including matching a vendor identification code with a subset of said collection of catalogs, wherein said subset of catalogs includes both a vendor catalog from a predetermined vendor and a second catalog from a predetermined third party;

a search program, said search program relying on said second set of criteria to select specific items from said catalogs determined from said catalog selection protocol; and

a cross-reference table linking a vendor item catalog number from said vendor catalog with an item catalog number from said predetermined third party.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO : 6,055,516

DATED : April 25, 2000

INVENTOR(S) : James M. Johnson et al.

It is certified that error appears in the above-identified patent and that said Letters Patent are hereby corrected as shown below:

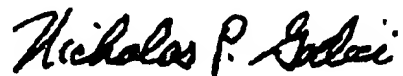
On title page, item 73 Assignee
replace "Procurenent, Inc."
with --Fisher Scientific Company L.L.C.--.

On title page, item 63 Related U.S. Application Data
please insert --, issued as US Patent No. 6,023,683 on February 8, 2000--.

Col. 1, line 2
please insert --, issued as US Patent No. 6,023,683 on February 8, 2000--.

Signed and Sealed this

First Day of May, 2001



Attest: .

NICHOLAS P. GODICI

Attesting Officer

Acting Director of the United States Patent and Trademark Office



US006115641A

United States Patent [19]**Brown et al.**[11] **Patent Number:** **6,115,641**[45] **Date of Patent:** **Sep. 5, 2000**

[54] **SYSTEMS AND METHODS FOR FACILITATING THE EXCHANGE OF INFORMATION BETWEEN SEPARATE BUSINESS ENTITIES**

5,548,518 8/1996 Dietrich et al. 700/100
5,694,546 12/1997 Reisman 705/26

FOREIGN PATENT DOCUMENTS

0466 089 A2 1/1992 European Pat. Off. .
2 293 902A 4/1996 United Kingdom .
WO 95/33236 12/1995 WIPO .

OTHER PUBLICATIONS

Brown, Keith T., *The Builder's Revolution*, pp. 185-257, ISBN: 0-9649339-X, Sun Forest systems Publishing (1995).

Leuthold, Jrm, *CPM scheduling is key to project efficiency; critical path method; Construction Management Column*, vol. 69, No. 5, p. 43, ISSN: 0033-4801, Pulp & paper, Miller Freemans Inc. (1995).

Valigra, Lori, *Saturn's NT Effect*, vol. 3, No. 6, p. 040, Client/Sewer Computing, Senting Publishing Co. (1996).

Primary Examiner—William Grant

Assistant Examiner—Zoila Cabrera

Attorney, Agent, or Firm—Myers Bigel Sibley & Sajovec, P.A.

[57]

ABSTRACT

A computer based product catalog system is provided for automatically distributing product information to members of a network. Electronically stored catalogs of multiple manufacturers' products, including descriptions of the products, are provided. An electronically stored distributor catalog, including selected products from the various manufacturers is also provided. Descriptions of the products in the manufacturers' catalogs may be updated automatically. In response to the updating of the manufacturers' catalogs, the descriptions of the products in the distributor's catalog may be automatically updated in real time. A terminal located at the retail level of distribution may be operatively connected to the manufacturers' catalogs and to the distributor catalog to allow viewing of product information therewithin. The terminal may enable users to order products from a manufacturer's and distributor's catalog.

2 Claims, 11 Drawing Sheets**Related U.S. Application Data**

[62] Division of application No. 08/775,276, Dec. 31, 1996, Pat. No. 5,923,552.

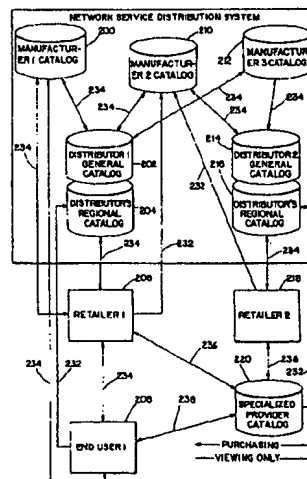
[51] Int. Cl.⁷ G06F 19/00; G06F 17/60

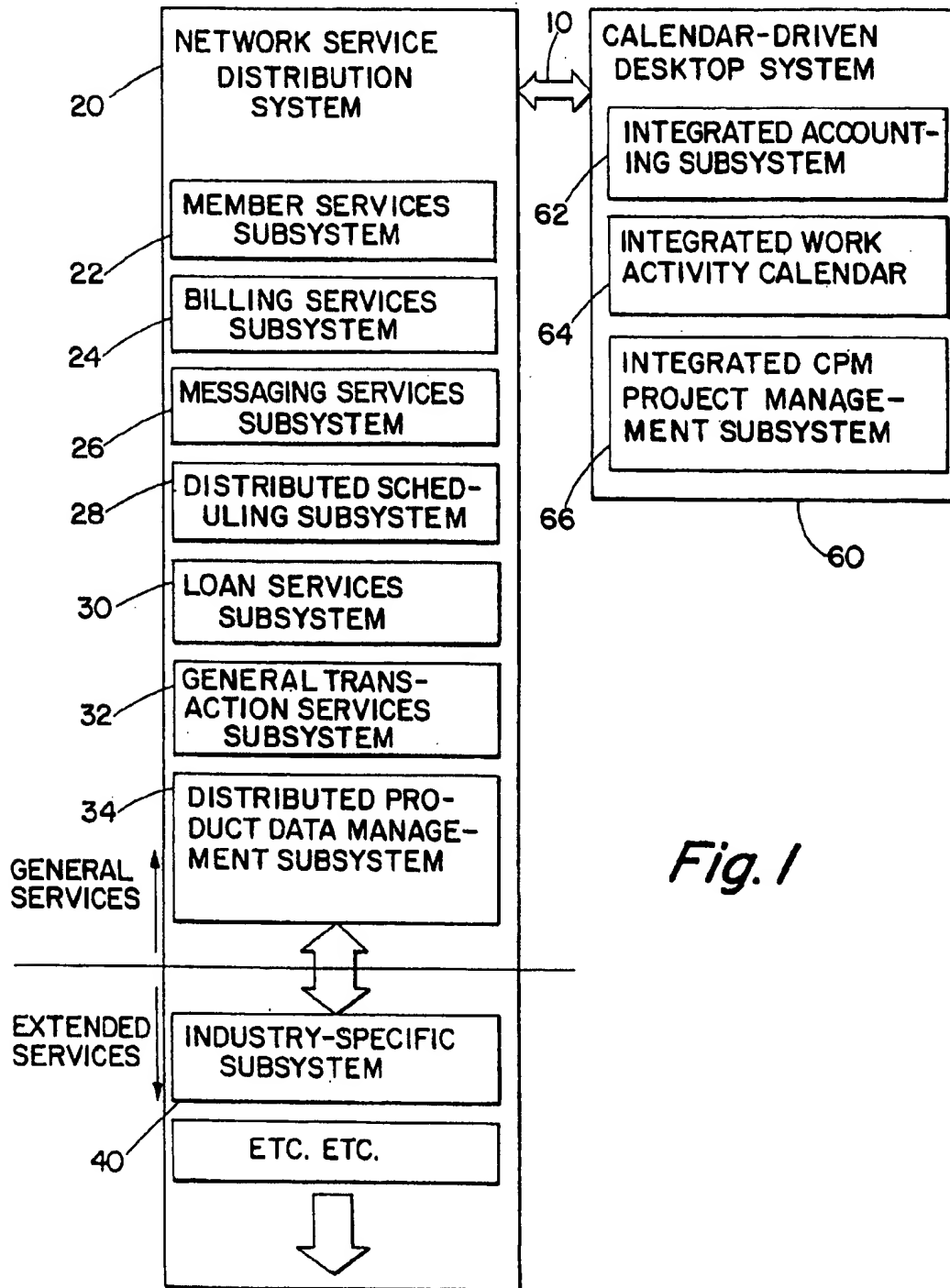
[52] U.S. Cl. 700/102; 705/27; 705/26;
705/28

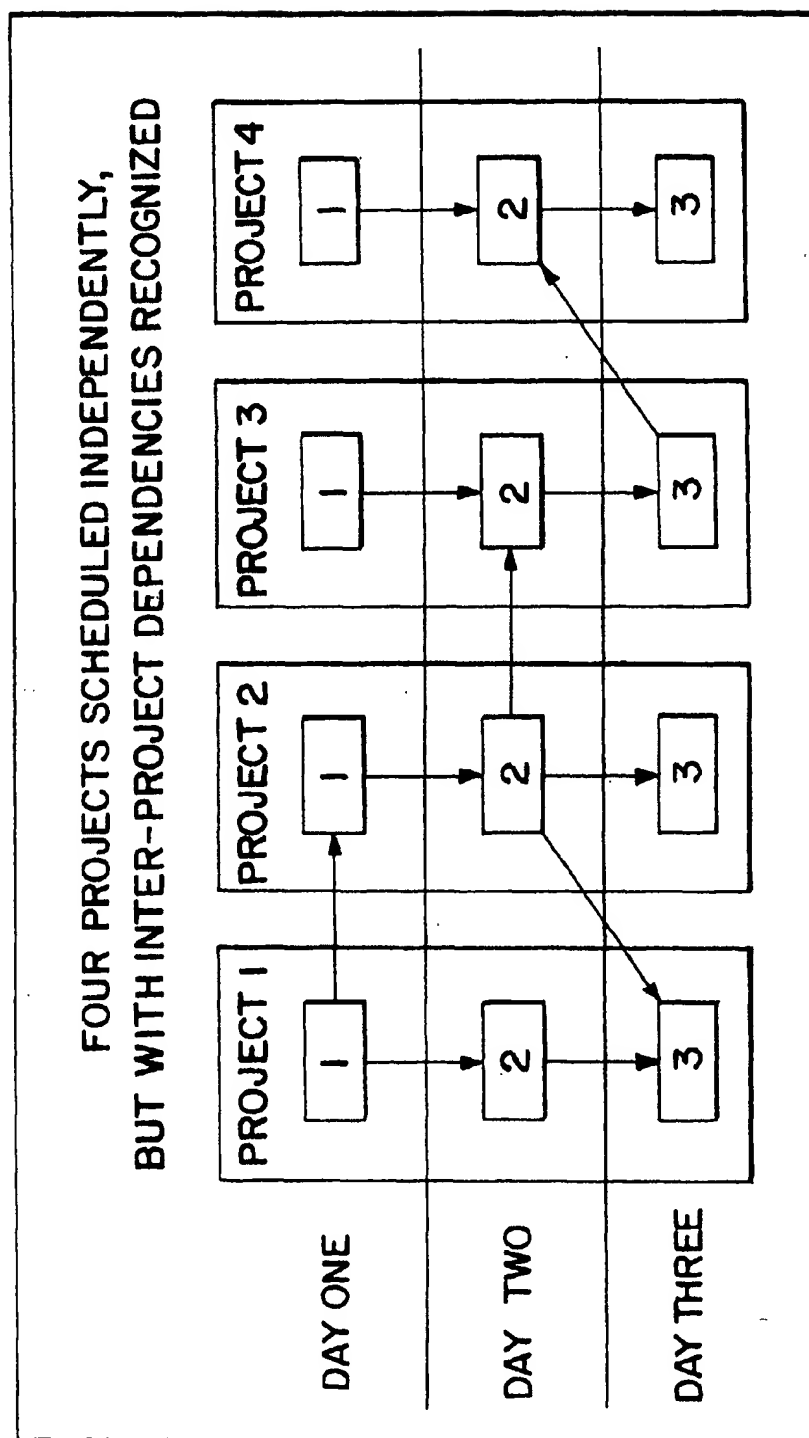
[58] Field of Search 705/27, 26, 28;
700/102

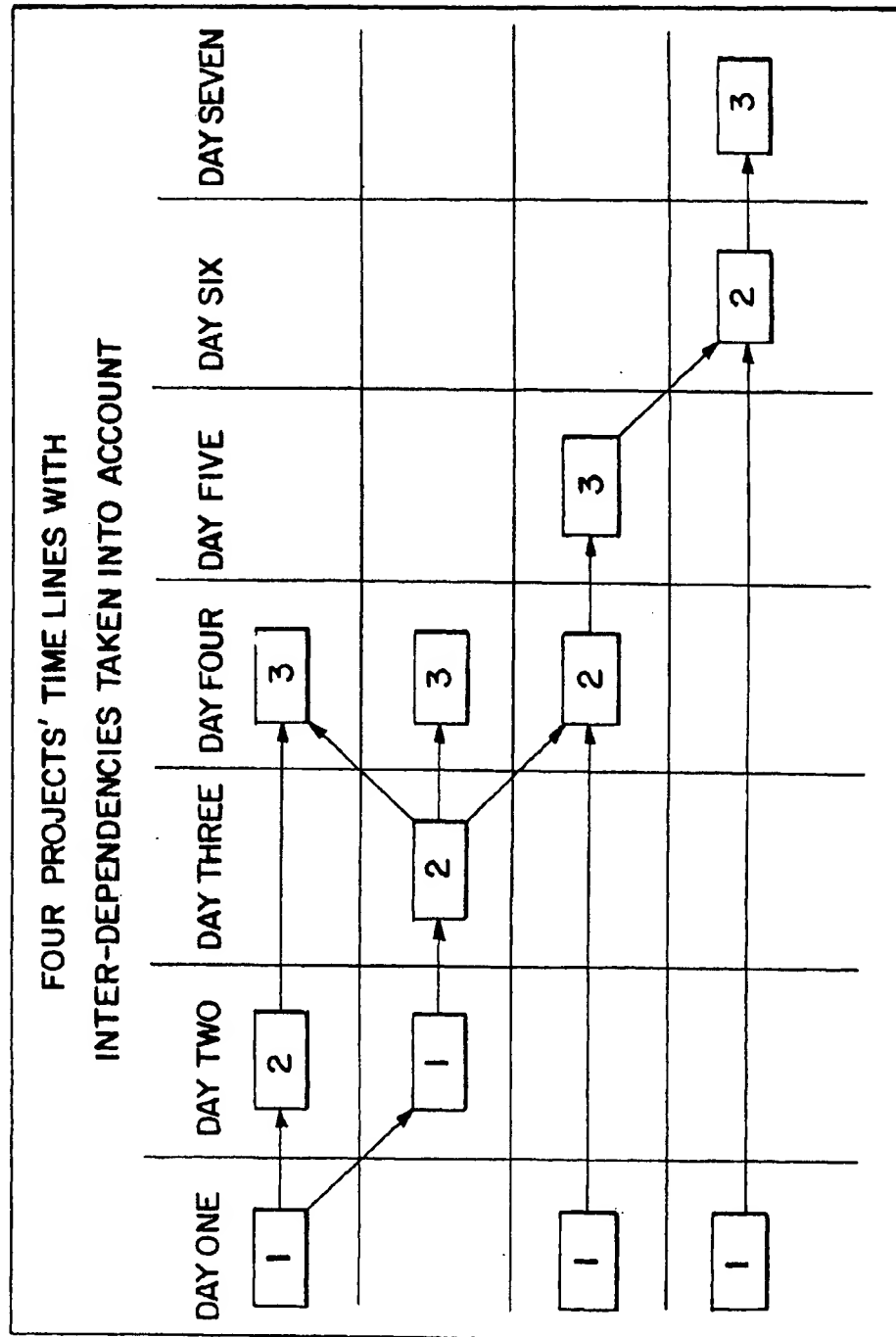
[56] **References Cited****U.S. PATENT DOCUMENTS**

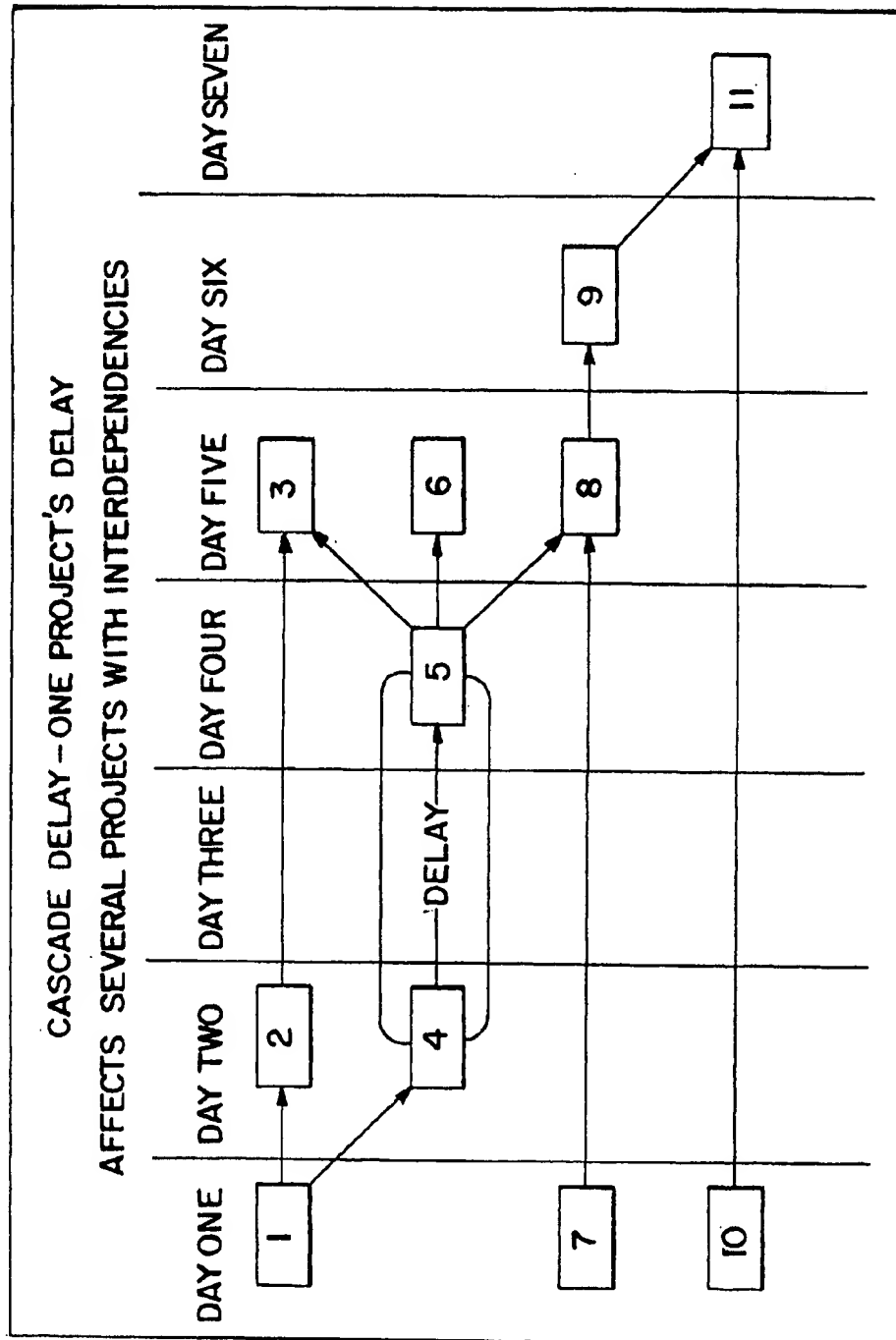
4,700,318	10/1987	Ockman	345/431
4,852,001	7/1989	Tsushima et al.	705/8
4,937,743	6/1990	Rassman et al.	705/8
5,053,970	10/1991	Kurihara et al.	700/104
5,233,533	8/1993	Edstrom et al.	700/103
5,255,181	10/1993	Chapman et al.	705/8
5,283,865	2/1994	Johnson	345/357
5,319,542	6/1994	King, Jr. et al.	705/27
5,367,627	11/1994	Johnson	345/357
5,369,570	11/1994	Parad	705/8
5,381,332	1/1995	Wood	705/8
5,406,476	4/1995	Deiziel, Jr. et al.	705/8
5,450,317	9/1995	Lu et al.	705/10
5,467,268	11/1995	Sisley et al.	705/9
5,487,144	1/1996	Takahashi et al.	345/348
5,493,490	2/1996	Johnson	705/26
5,548,506	8/1996	Srinivasan	705/8



*Fig. 1*

*Fig. 2A*

*Fig. 2B*

*Fig. 2C*

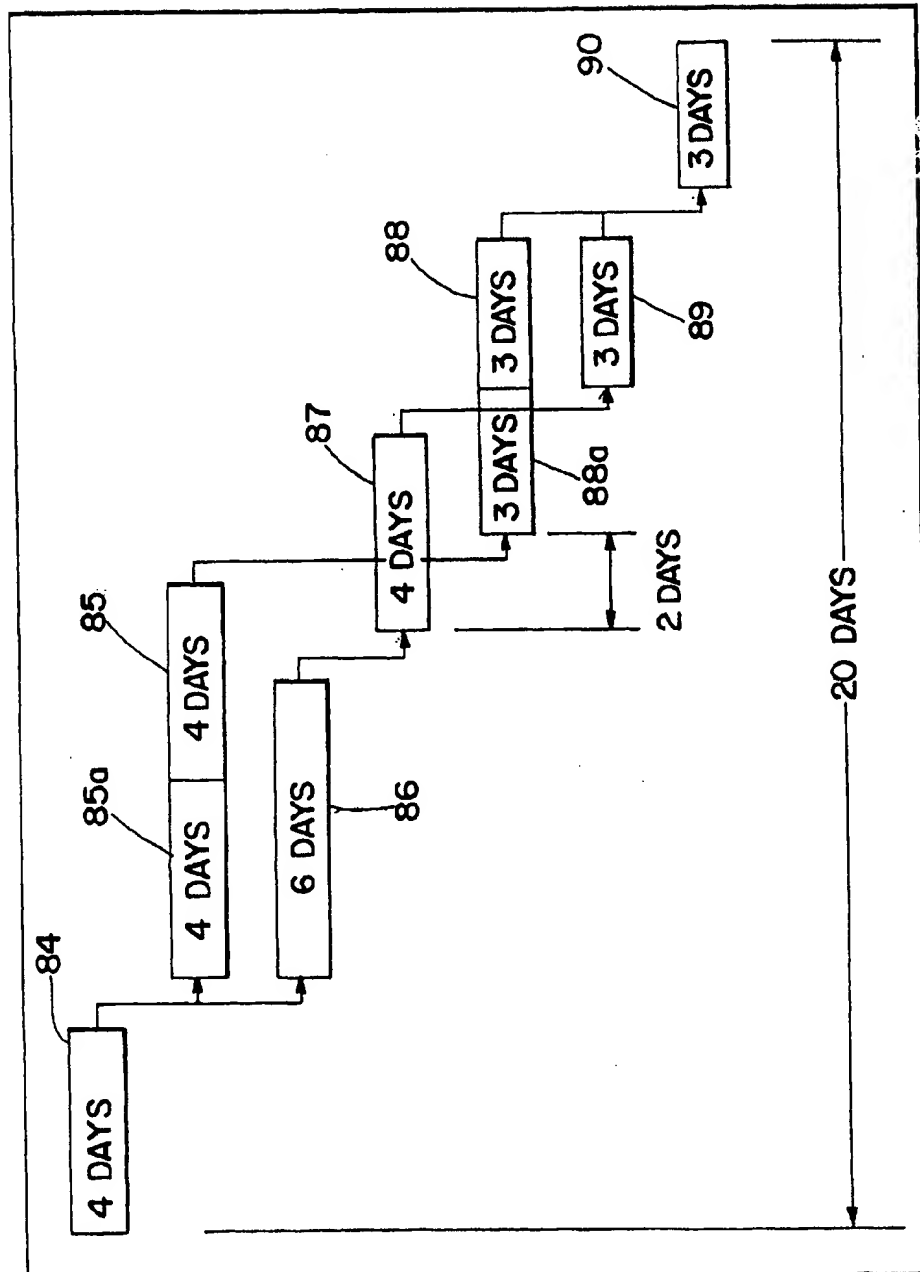


Fig. 3A

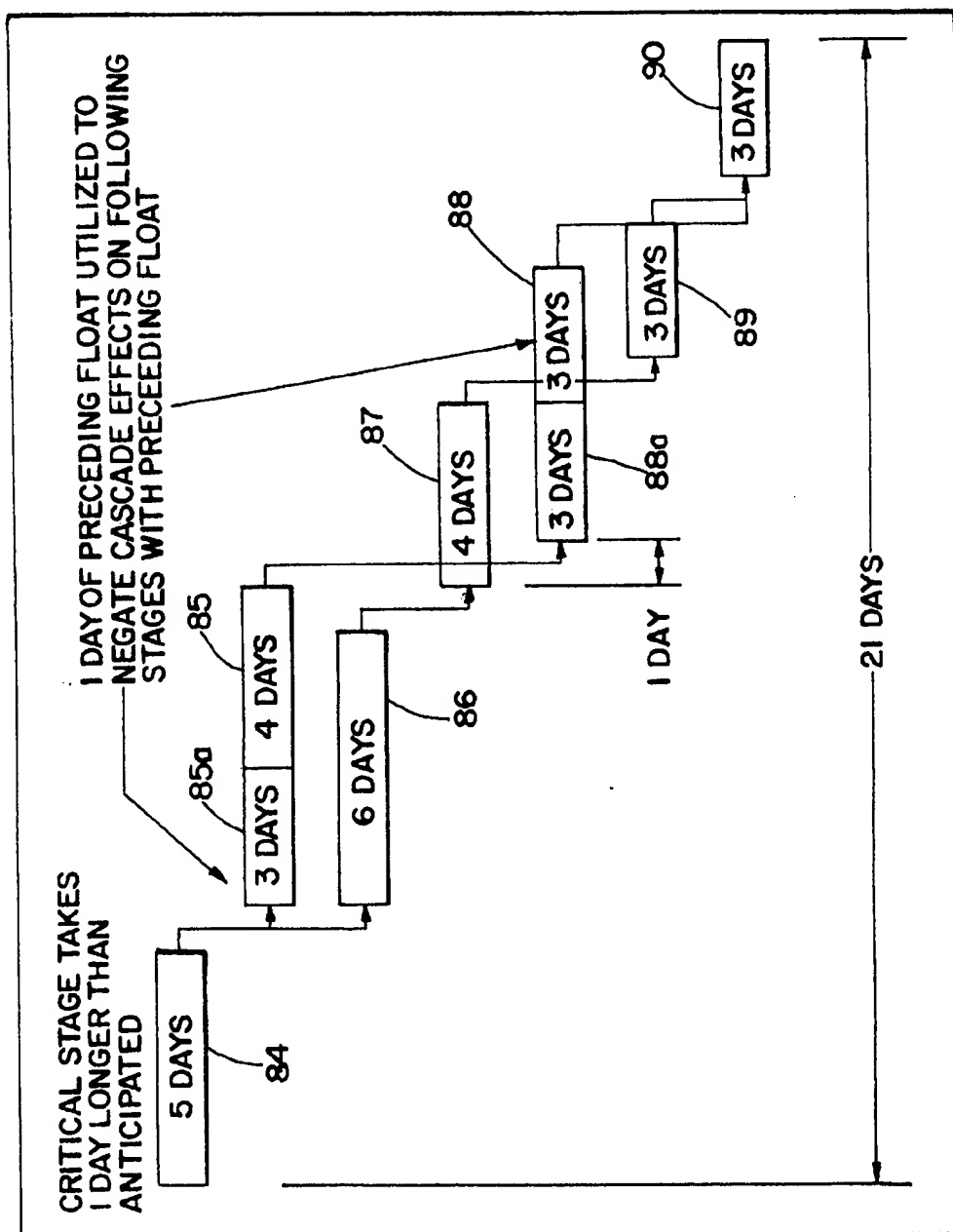


Fig. 3B

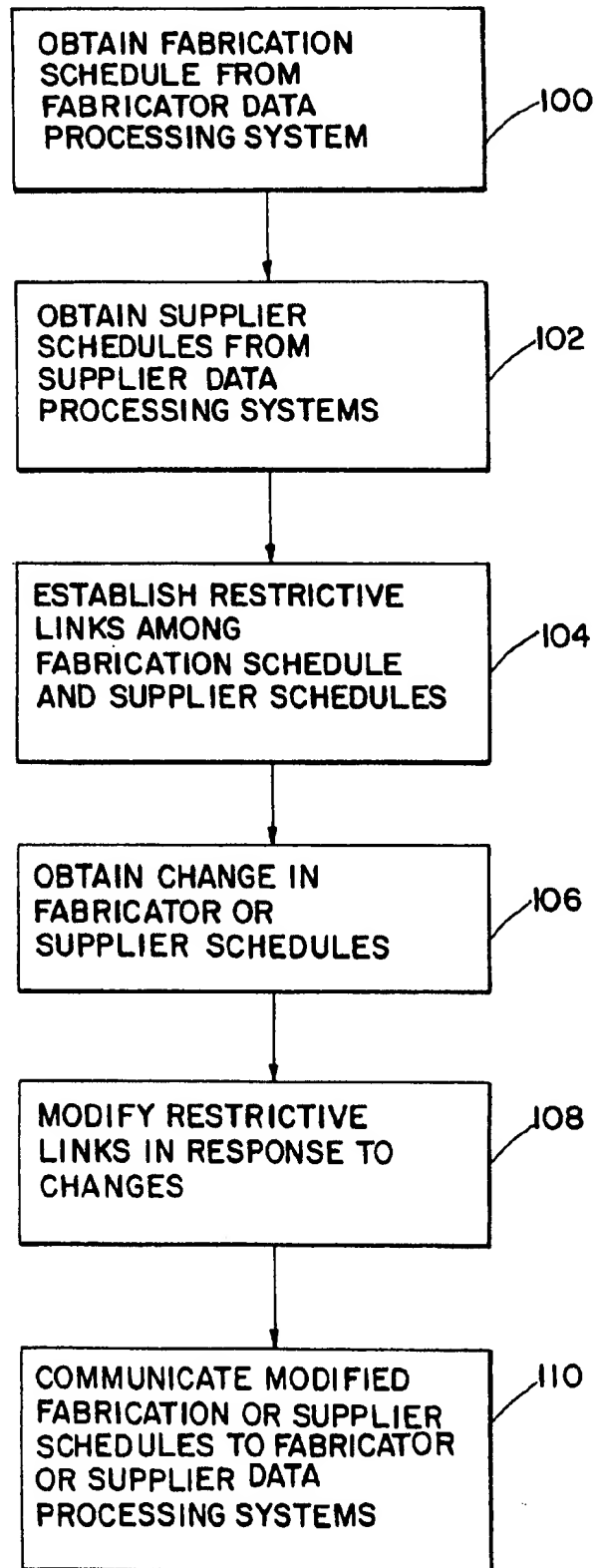
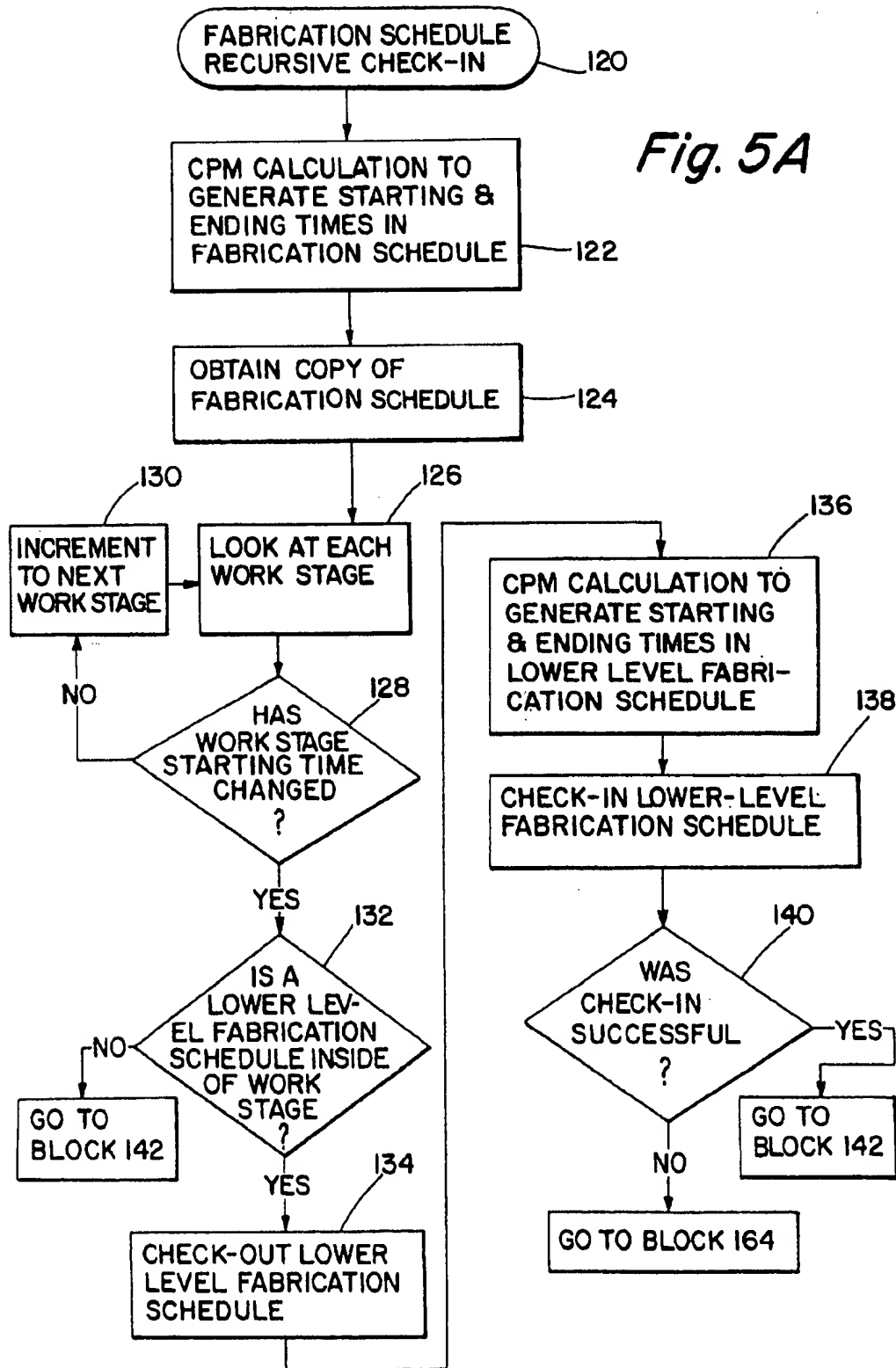
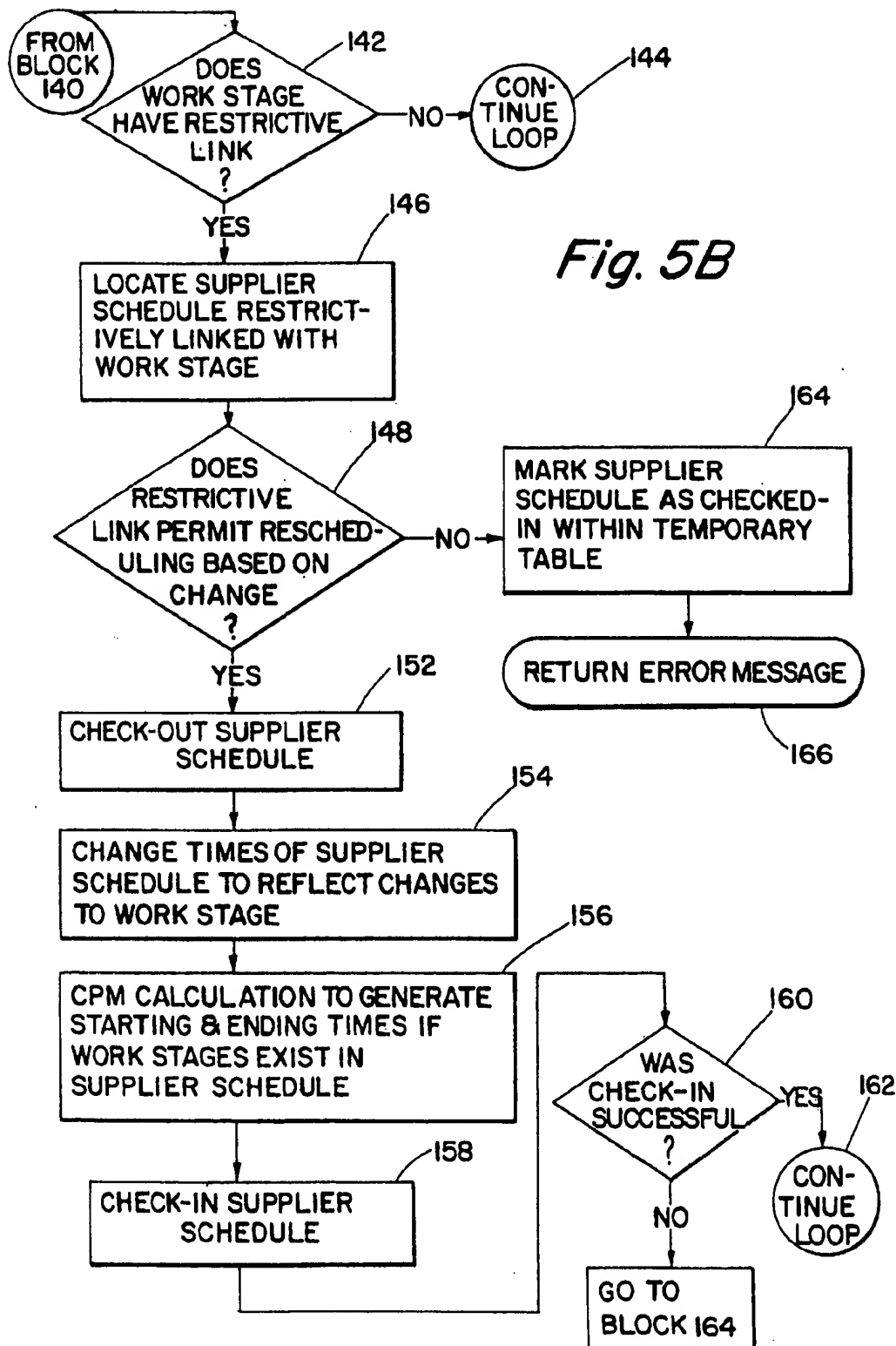
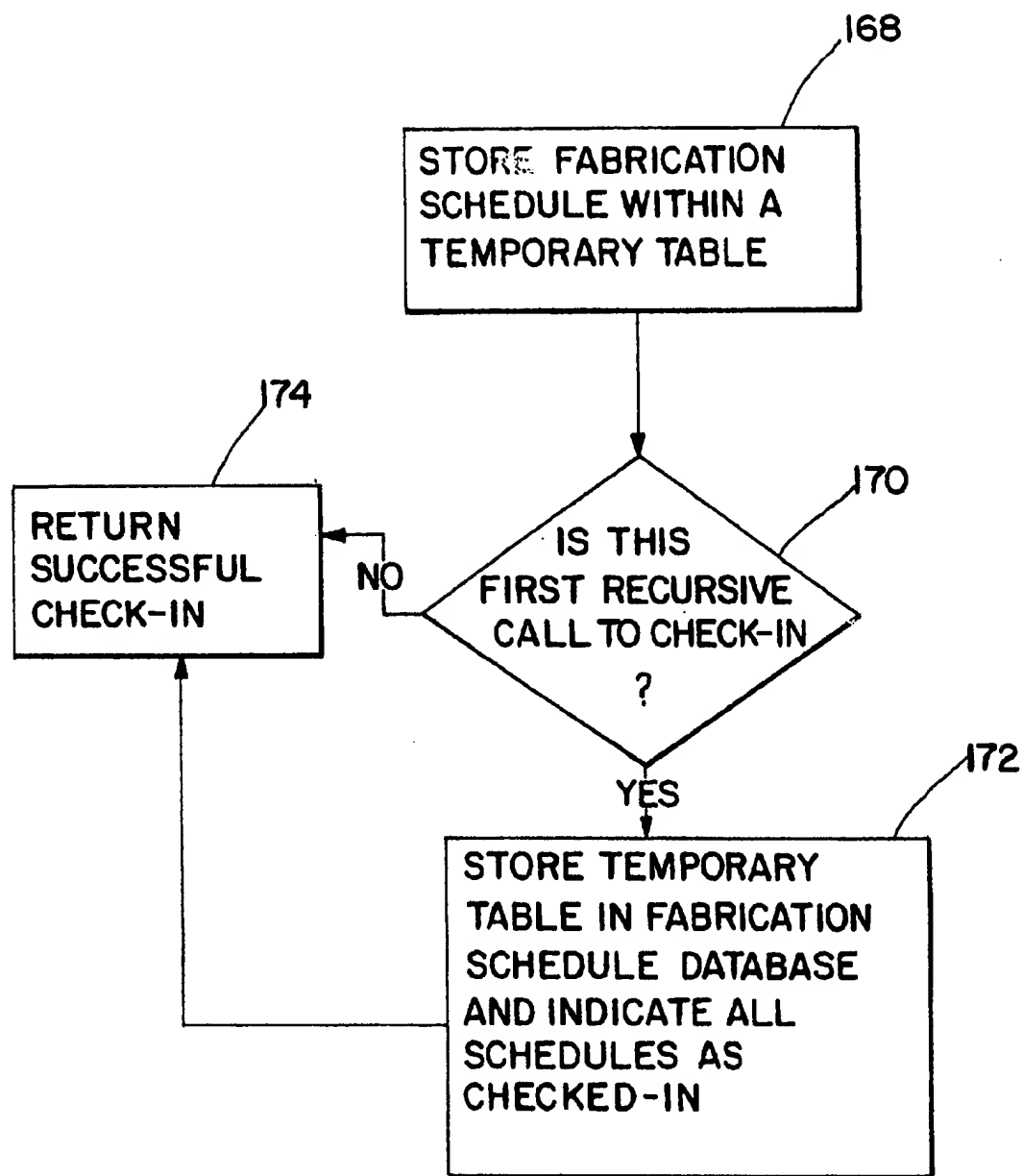
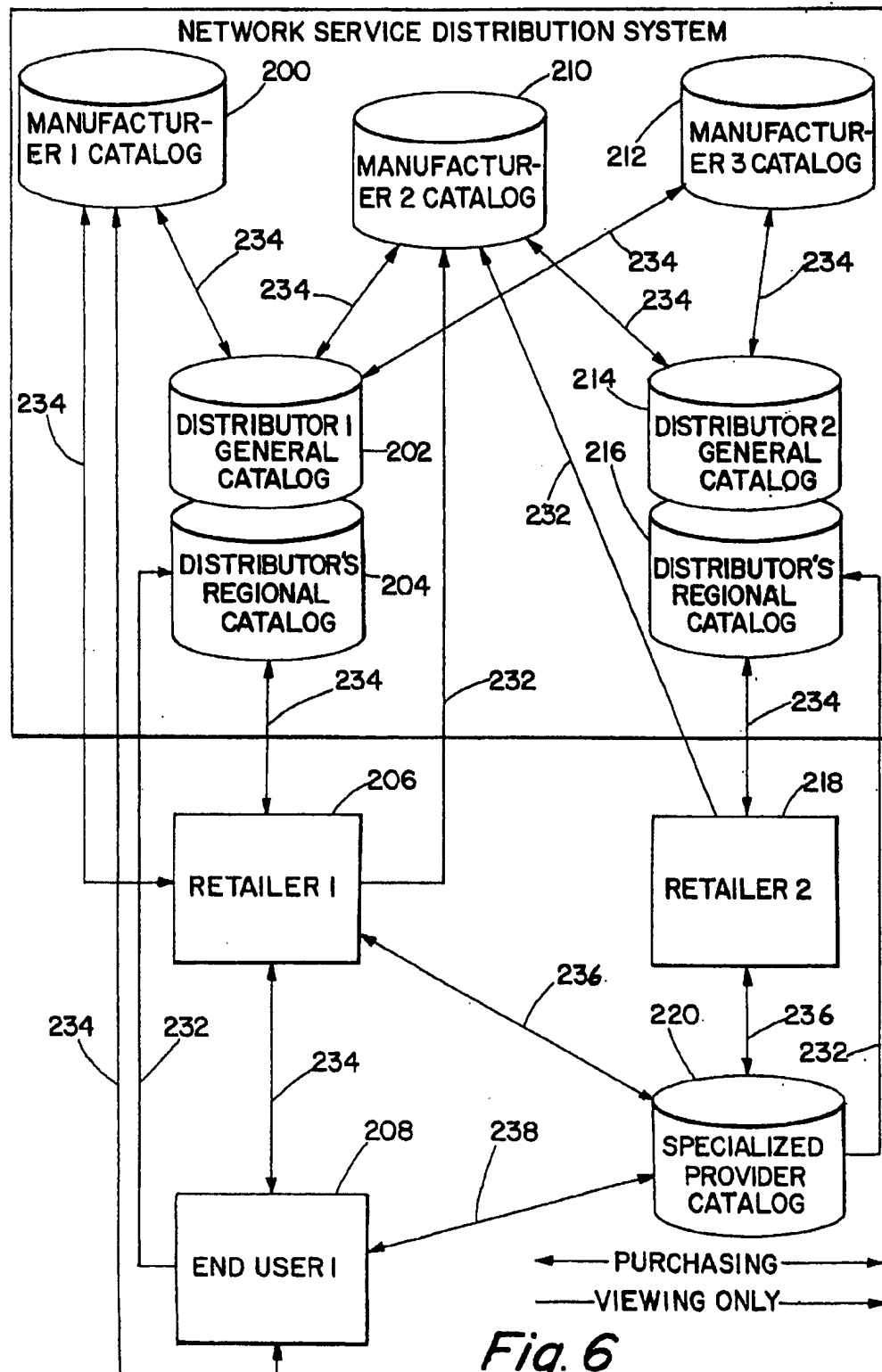
Fig. 4

Fig. 5A



*Fig. 5C*



SYSTEMS AND METHODS FOR FACILITATING THE EXCHANGE OF INFORMATION BETWEEN SEPARATE BUSINESS ENTITIES

CROSS REFERENCE TO RELATED APPLICATION

This application is a divisional of application Ser. No. 08/775,276, filed Dec. 31, 1996, now U.S. Pat. No. 5,923,552.

FIELD OF THE INVENTION

The present invention relates generally to electronic communications and more particularly to electronic communications via computer networks.

BACKGROUND OF THE INVENTION

Electronic communications via computer networks are replacing traditional modes of communication including printed media and voice communication. One form of computer network communication, known as Electronic Data Interchange (EDI), is becoming a standard format for exchanging information within a business. However, the use of network communications such as EDI as a means for communicating between separate business entities has not received widespread acceptance because of the independent nature of the computer networks of separate business entities.

Computer networks have also been used within a business to implement various quantitative analysis techniques such as Critical Path Method (CPM) scheduling, Project Evaluation and Review Technique (PERT) and Material Requirement Planning (MRP). These tools are used to increase the efficiency of business operations and to coordinate work flow within a business. For example, U.S. Pat. No. 5,233,533 to Edstrom et al. describes scheduling software for optimizing the scheduling of resources in a factory setting. U.S. Pat. No. 4,937,743 to Rassman et al., describes a method for the dynamic management of a project involving multiple interrelated resources, such as the use of facilities and equipment within a business entity. Unfortunately, the independent nature of many computer networks and the programs designed to run on these computer systems, has made the coordination of schedules between separate business entities difficult. Consequently, computer systems have often not been used to facilitate the scheduling of interrelated operations of individual business entities across an entire industry.

Product information is often transmitted by manufacturers to distributors in an electronic format. Because product distributors typically sell products from multiple manufacturers, each distributor typically develops its own catalog of products using the information provided by the various manufacturers. As a result, updated product information received from a manufacturer is not easily or promptly forwarded to retailers and others who purchase products from distributors because the distributor must update its catalog.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to facilitate communication and information exchange between independent business entities.

It is another object of the present invention to facilitate the coordination of schedules of individual business entities within an industry.

It is another object of the present invention to facilitate prompt dissemination of product information from manufacturers to others downstream in the distribution chain.

The present invention includes systems, methods, and computer program products for synchronizing fabrication schedules and supplier schedules. A fabrication schedule includes a plurality of sequential work stages which are stored on a fabricator data processing system. Supplier schedules include a plurality of schedules for the sequential work stages within a fabrication schedule. Supplier schedules may be stored on supplier data processing systems. A fabrication schedule is obtained from a fabricator data processing system, and supplier schedules are obtained from respective supplier data processing systems. Restrictive links are established between the fabrication schedule and the supplier schedules. Each restrictive link defines the supplier that will perform a work stage, and may also define the starting and ending times for both fabrication and supplier schedules.

When a change in at least one of the sequential work stages is obtained from the fabricator or from the selected one of the suppliers, the restrictive links are automatically modified in response to the obtained change. The modified fabrication schedule and/or the modified supplier schedule is communicated to the fabricator data processing system or to the supplier data processing system. If a restrictive link cannot be modified in response to an obtained change, an error message may be returned. Furthermore, if a supplier is not able to supply a particular work stage, a second supplier may be automatically selected.

The present invention facilitates synchronizing relevant portions of a product fabricator's schedules with the various schedules of suppliers of materials, labor, and the like. The present invention also facilitates the synchronization of accounting and billing systems, access to product-related information, access to legislative and regulatory information, and access to on-line catalogs and ordering systems for various materials.

According to another aspect of the present invention, a scheduling method is provided for reducing the time to complete a critical path schedule when completion of at least one of the activities in the critical path is delayed. A schedule having interrelated activities forming a critical path is stored in a data processing system. A float time preceding a selected activity starting time is assigned and utilized to absorb delays in completing activities preceding the selected activity. A revised schedule may be generated in the data processing system based upon the absorbed delays. Float time may be assigned to multiple selected activities.

According to another aspect of the present invention, a computer based product catalog system is provided for automatically distributing product information. Electronically stored catalogs of multiple manufacturers' products, including descriptions of the products, are provided. An electronically stored distributor catalog, including selected products from the various manufacturers is also provided. Descriptions of the products in the manufacturers' catalogs may be updated automatically. In response to the updating of the manufacturers' catalogs, the descriptions of the products in the distributor's catalog may be automatically updated. A terminal located at the retail level of distribution may be operatively connected to the manufacturers' catalogs and to the distributor catalog to allow viewing of product information therewithin. The terminal may enable users to order products from a manufacturer's and distributor's catalog.

The present invention is advantageous because it facilitates the coordination and flow of information between

separate business entities within an industry. In particular, schedules of separate business entities can be synchronized to establish efficiencies across the industry. The present invention enhances the flow of information among businesses without disrupting the autonomy of each business. Furthermore, the present invention allows information to flow between the members of an industry without interfering with the established channels of trade and business relationships.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a Network Service Distribution System and a Calendar-Driven Desktop System for synchronizing schedules and exchanging information between separate business entities, according to the present invention.

FIGS. 2A, 2B, 2C illustrate the interdependencies of multiple projects.

FIGS. 3A, 3B illustrate adding preceding float time to various work stages in a CPM schedule, according to the present invention.

FIG. 4 schematically illustrates a method of synchronizing a fabrication schedule and a plurality of supplier schedules, according to the present invention.

FIGS. 5A, 5B, 5C schematically illustrate in greater detail a method of synchronizing a fabrication schedule and a plurality of supplier schedules.

FIG. 6 illustrates a computer based product catalog system for distributing product information, according to the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention now is described more fully hereinafter with reference to the accompanying drawings, in which preferred embodiments of the invention are shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art.

According to one aspect of the present invention, a computer network system having a dynamic scheduling system for integrating schedules of separate business entities within an industry is disclosed. The present invention leaves the independence and management of individual business entities intact, yet allows the whole industry to work as a single, efficient organization. Utilizing the present invention, a business entity may view and interact with information of other members an entire industry. Each business continues to compete with other businesses in the industry, yet scheduling of activities, project management and exchange of information necessary for the whole industry to function can become organized and efficient via the present electronic computer network system. To facilitate scheduling and project management across an entire industry, the present invention utilizes and enhances known scheduling techniques including Critical Path Method (CPM) to allow each network member to maintain individual schedules yet synchronize relevant portions of these schedules with the schedules of other industry members.

Using the home building industry as an illustrative example, the present invention facilitates synchronizing relevant portions of a home builder's schedules with the various schedules of vendors, suppliers, manufacturers,

financial institutions, and the like. The present invention also facilitates the synchronization of accounting and billing systems, access to home building-related information, access to legislative and regulatory information, and access to on-line catalogs and ordering systems for materials, fixtures, and the like.

For each individual house being fabricated by a home builder, the home builder typically will develop a fabrication schedule which includes a sequential listing of all activities or work stages related to the completion and sale of the house. For example, work stages may include, but are not limited to, grading the lot upon which a house is to be built, digging the foundation, pouring the footings upon which the house rests, framing the structure, placing a roof over the framing, covering the framing with sheathing, laying up brick veneer, and finishing the inside of the house. Each of these work stages will typically be assigned starting times and will be sequentially arranged so that the house is built in the shortest amount of time. As is known to those having skill in the art, some work stages may occur substantially concurrently, while others require the completion of a preceding work stage before they can occur.

The home builder will typically enter into a relationship or contract with another party, such as a sub-contractor or material supplier to perform the work and/or deliver materials for each of these work stages. In either case, the contracting party is a supplier of either labor or materials or both. Each contract between the home builder and a supplier of labor/materials typically includes a time of performance on the part of the supplier. That is, each contract will restrictively link the supplier's schedule of performance to the home builder's schedule for a particular house. Because most suppliers of labor/materials are supplying labor/materials for multiple houses and projects, it is important that a home builder's schedule for a particular house be synchronized with a supplier's schedule. In particular, it is important for a home builder to know that if a work stage starting date is changed for some reason, the supplier supplying labor/materials for that work stage either can or cannot perform under the contract. If the supplier cannot perform because of its schedules related to other projects, the home builder must secure a contract with another supplier.

An exemplary work stage during the construction of a house is the framing of the house structure. A home builder will typically locate a particular supplier (i.e. a framing crew) to begin and complete the framing work stage within a given time period. For example, assume that framing of a particular house is slated to take twenty days beginning on January 1. The home builder enters into a contract with framing crew A to begin framing on January 1 and complete framing no later than January 21. Typically, the contract or restrictive link between the home builder and framing crew A will include some reasonable time range in which the starting date of framing can vary, such as plus or minus three days. Thus, if the foundation work, which must be completed prior to framing, is not completed until January 2, the home builder knows that framing crew A can still perform. However, if the foundation work is not completed until January 18, the contract between the home builder and framing crew A is no longer valid and the home builder must either ask framing crew A if it can still perform, or the home builder must look for another framing crew.

The above scenario is often repeated in the home building industry and other fabrication industries many times during a single project. The home builder or other product fabricator typically spends a lot of time securing contracts with

suppliers of labor/materials, both at the beginning of a fabrication schedule and throughout the schedule as the starting and ending times of the various work stages are changed. The present invention facilitates establishing contracts or restrictive links between product fabricators, such as home builders, and suppliers of labor and/or materials. The present invention also facilitates the automatic synchronization of a fabricator's schedule for a particular project or product with the schedules of many suppliers of labor and/or materials as changes occur to either the fabricator's schedule or a supplier's schedule or both. The present invention also facilitates rescinding a contract between a fabricator and a supplier when the rescinded contract does not permit a particular change to either the fabrication schedule or the supplier's schedule, and facilitates the establishment of a replacement contract between the fabricator and another supplier. The present invention may be utilized in a variety of industries and is not limited to the home building industry.

A preferred computer network system for carrying out methods of synchronizing a fabrication schedule with a plurality of supplier schedules, according to the present invention, is a dynamic system wherein schedule changes made by a network member ripple down to all network members and are automatically integrated within the schedule of each respective network member as appropriate. For example, if rain delays the framing of a house, the lumber yard's delivery schedule will be updated automatically to deliver lumber on a new date specified in the home builder's schedule. The present invention permits such changes to be made remotely through the use of various input devices, including personal data assistants and other computer terminals in communication with a central data processing system.

Referring now to FIG. 1, a computer network system 10 for synchronizing schedules and facilitating the flow of information between multiple business entities within an industry, according to a preferred embodiment of the present invention, is illustrated. In the illustrated embodiment, the computer network system 10 includes a centrally-located Network Service Distribution System 20 and a plurality of Calendar-Driven Desktop Systems 60, each in communication with the Network Service Distribution System. Each Calendar-Driven Desktop System 60 is the user interface members of the computer network system 10 use to communicate and exchange information with other network members via the Network Service Distribution System 20. The Network Service Distribution System 20 and each Calendar-Driven Desktop System 60 contain subsystems that are described in detail below.

Network Service Distribution System

The Network Service Distribution System 20 contains the following subsystems that may apply generally to any industry: Member Services Subsystem 22, Billing Services Subsystem 24, Messaging Services Subsystem 26, Distributed Scheduling Subsystem 28, Loan Services Subsystem 30, General Transaction Services Subsystem 32, Distributed Product Data Management Subsystem 34, and a Consumer Product or Service Marketing Subsystem (not shown). These subsystems can be categorized as general services, as illustrated, because they are typically useful and important to any industry. In addition, one or more, industry-specific subsystems 40 can be added as necessary. Industry-specific subsystems 40 can be categorized as extended services, as illustrated, because they are typically unique to a particular industry.

The Member Services Subsystem 22 and the Billing Services Subsystem 24 are used to maintain information

about network members. The term "network members" shall mean business entities within an industry that have access to the computer network system 10 illustrated in FIG. 1, or a computer network system capable of carrying out the methods of the present invention. It is contemplated by the present invention that each network member is a distinct and separate business entity within a particular industry. The Member Services Subsystem 22 preferably maintains the list of network members and validates the identity of a user in communication with the Network Service Distribution System 20. The Billing Services Subsystem 24 preferably tracks the usage of the Network Service Distribution System 20 for each network member and generates billing information for each network member. The Messaging Services Subsystem 26 is used to store and forward electronic mail and other asynchronous message types between network members. Preferably, various publications and discussion forums are accessible by network members via the Messaging Services Subsystem 26. Also, gateways are preferably provided to the Internet and other electronic networks for inter-industry information exchange. The Distributed Scheduling Subsystem 28, described in greater detail below allows each network member to integrate its schedules within the schedules of other network members.

The Loan Services Subsystem 30 facilitates access to financial services for network members and their customers. Preferably, communication with major financial institutions is available via the Loan Service Subsystem 30. The General Transaction Services Subsystem 32 facilitates a store and forward process for standardized messages that routinely travel between network members in the course of conducting commerce. Exemplary messages include standardized accounting messages and standardized project management messages. Preferably, all messages passing through are stored for audit trail purposes. The Distributed Product Data Management Subsystem 34 makes available electronically recorded product information to distributors, retailers, and consumers. The Consumer Product or Service Marketing Subsystem (not shown) connects network members with consumers and facilitates "on-line" shopping.

The Network Service Distribution System 20 and its illustrated subsystems may serve as a means for: obtaining a fabrication schedule from a fabricator's data processing system; obtaining supplier schedules from respective supplier data processing systems; automatically selecting a supplier from a plurality of suppliers; and communicating a selection to the supplier system which corresponds to the selected supplier. The Network Service Distribution System 20 may also include a fabricator data processing system for storing fabrication schedules which include multiple work stages arranged sequentially. The Network Service Distribution System 20 may also include a plurality of supplier data processing systems for storing a schedule for each sequential work stage performed by a particular supplier.

Calendar-Driven Desktop System

The Calendar-Driven Desktop System 60 provides the network interface for communications with the Network Service Distribution System 20 and with other network members. It also provides users with a calendar of tasks and work stages needed for carrying out the operations of a network member's business. Standardized messages from the General Transaction Services Subsystem 32 can automatically produce intelligent calendar entries. These calendar entries may be employed to launch applications when selected, or they can automatically launch on a given date. Calendar entries can likewise be used to send standardized

messages back to the General Transaction Services Subsystem or can send them automatically 32.

Using the home building industry as an example, when it is time to order supplies for a given work stage of a project, a calendar entry created by a project management or project planning application can automatically place the order. Calendar entries can also automatically confirm that an order will be delivered on time. Recurring calendar entries can be used to launch recurring applications or functions appropriate for a network member's business. Calendar entries may be used to integrate project management, integrate work activity calendars, and integrate accounting. For project management, calendar entries can issue purchase orders and bid requests, or ask for confirmation regarding material delivery. For accounting, calendar entries can be used to send and receive invoices. Any subsystem application can be written under the Calendar-Driven Desktop System 60 to provide that particular application with industry-wide communication through the Network Service Distribution System 20 utilizing a standardized application programming interface (API).

In the illustrated embodiment of FIG. 1, the Calendar-Driven Desktop System 60 includes the following: Integrated Accounting Subsystem 62, Integrated Work Activity Calendar 64, and Integrated Project Management Subsystem 66. The Calendar-Driven Desktop System 60 receives and sends messages between network members and the various subsystems within the Network Service Distribution System 20. Each message is received and distributed within the Calendar-Driven Desktop 60 according to the subsystem or program application most appropriate to its purpose. For example, messages from the General Transaction Services Subsystem 32 arrive as stages of work or tasks linked to the Integrated Accounting Subsystem 62. These messages appear as calendar entries requiring attention by a network member. Preferably, calendar entries are automatically entered and adjusted by the Integrated Accounting Subsystem 62 and Integrated Project Management Subsystem 66. Calendar entries or messages from the General Transaction Services Subsystem 32 may automatically launch appropriate applications associated with completing or resolving the specific type of standardized message.

The Calendar-Driven Desktop System 60, and its illustrated subsystems, may serve as a means for: obtaining a fabrication schedule from a fabricator's data processing system; obtaining supplier schedules from respective supplier data processing systems; automatically selecting a supplier from a plurality of suppliers; and communicating a selection to the supplier system which corresponds to the selected supplier. The Calendar-Driven Desktop System 60 may also include a fabricator data processing system for storing fabrication schedules which include multiple work stages arranged sequentially. The Calendar-Driven Desktop System 60 may also include a plurality of supplier data processing systems for storing a schedule for each sequential work stage performed by a particular supplier.

The Integrated Project Management Subsystem 66 is a network member's interface with the Network Service Distribution System 20 for fabrication and supplier project schedules, such as CPM schedules. Using the home building industry as an example, a home builder may use the Integrated Project Management Subsystem 66 to obtain, modify, and communicate to suppliers its fabrication schedule. Similarly, each supplier of labor and/or materials may use the Integrated Project Management Subsystem 66 to obtain, modify, and communicate its schedules. The Integrated Work Activity Calendar 64 is a network member's interface

with the Network Service Distribution System 20 for work activity calendar schedules. The Integrated Project Management system 66 and the Integrated Work Activity Calendar 64 can both generate bid requests to other network members via the General Transaction Services Subsystem 32 for the purpose of establishing contracts between network members. If bid requests are accepted, contracts can be established using the General Transaction Services Subsystem 32. Once a contract is established, the Distributed Scheduling Subsystem 28 may serve as a means for establishing restrictive links between the schedules of the contracting parties. Network members may utilize the Calendar-Division Desktop System 60 on an on-going basis to monitor and update work progress on particular projects. When network members indicate that a work stage or task is complete, the restrictive link may be removed by the Distributed Scheduling Subsystem 28.

An Integrated Accounting Subsystem 62 is preferably integrated with project management and work activity schedules of a network member such that accounting information is automatically obtained without redundant data entry. Distributed scheduling and project management, along with the General Transaction Services Subsystem 32, allows the Integrated Accounting Subsystem 62 of the Calendar-Driven Desktop System 60 to function with little or no data entry. Accordingly, the majority of normal functions required for bookkeeping, such as invoice entry, may be eliminated.

The Integrated Work Activity Calendar Subsystem 64 is used by suppliers who enter into contracts with other network members to supply labor and/or materials for a work stage of a fabrication schedule. The Integrated Work Activity Calendar Subsystem 64 facilitates receiving bid requests for the supply of labor and/or materials from network members and is designed to automatically accept or reject bid requests, depending on the nature of the bid requests and also depending on the availability of time in a network member's schedule. Advantageously, since schedules are maintained by the Distributed Scheduling Subsystem 28, a network member does not have to be in continuous communication with the computer network system 10. A network member can, instead, periodically dial-in and connect to the Network Service Distribution System 20 using a personal computer and modem, including a small portable computer with a cellular modem.

The Integrated Work Activity Calendar Subsystem 64 may serve as a means for obtaining fabrication schedules and supplier schedules from the Distributed Scheduling Subsystem 28. If changes are made to a schedule, the Distributed Scheduling Subsystem 28 may serve as a means for obtaining the change and for automatically modifying a restrictive link associated with the change. If the restrictive link may not be modified because of schedule conflicts or for other reasons, an error message is returned. A network member has the option of canceling a contract with another network member, which deletes any restrictive links associated with the contract and sends a message to the affected network members, via the General Transaction Services Subsystem 32, indicating that a contract has been canceled. When an Integrated Project Management Subsystem 66 receives such a message it attempts to automatically handle the scheduling conflict. It issues bid requests to try and reestablish the contract with the same network member at a different time. If this fails within the preceding float range allowed for the conflicted work stage, the Integrated Project Management Subsystem 66 then issues bid requests to other network members. Unresolved scheduling conflicts preferably generate notifications to affected network members.

Schedule Synchronization Between Network Members

Typically each member of the computer network system 10 is a separate business entity within an industry and has one or more work schedules. Generally, there are two types of work schedules: CPM schedules, and work activity calendar schedules. A work activity calendar schedule is simply a schedule of appointments. It is similar like the notebook of scheduled activities that a businessman maintains. A CPM schedule is a sequential arrangement of work stages, some of which may not begin until previous ones have been completed. Individual tasks can be work stages in both types of schedules. Restrictive links can be established between either of these two types of schedules. Work stages within a schedule may have restrictions placed on them by their owners to indicate how far the starting and ending times of the work stage can vary.

A work stage is a step in the fabrication of a product and has a time duration defined by a starting time and an ending time. As used herein, the terms "time(s)" and "date(s)" shall have the same meaning and shall be interchangeable. Schedules can be at different scales, with more detailed schedules inside of lesser detailed schedules. For example: a schedule of new office construction in a business park—office one must be completed before office two can begin, and so forth. Each office project can have a detailed schedule of events required to complete its construction. Thus, the detailed schedule is a "child" of the lesser detailed "parent." Preferably, work stages cannot be linked to work stages shared between parent and child projects. For example, if work stage A is part of a project which is contained in work stage B, then no work stage that is in the same project as work stage B may be linked to work stage A, as this would cause a circular dependency. With this restriction in place, the system can allow for entire projects within a single work stage.

In the illustrated embodiment, the Distributed Scheduling Subsystem 26 automatically updates interrelated schedules of network members and allows each network member to make changes to its schedules in consideration of up-to-date knowledge about the status of the schedules of other network members. Preferably, the Distributed Scheduling Subsystem 26 utilizes enhanced CPM scheduling techniques which permit restrictions and relationships to be established between a broad range of work stages and project scales, each able to be restricted by the other, regardless of the scale of a stage of work. For example, a set of work stages that define a project can be considered as a single work stage that is related to or restricted by, one or more other work stages, jobs, large-scale projects, and the like. Similarly, a collection of jobs that make up a large scale project may be considered a single work stage that is related to or restricted by a work stage, a job, a large-scale project, and the like. Preferably, the Distributed Scheduling Subsystem 28 stores the various fabrication and supplier schedules of network members within one or more databases.

Referring now to FIG. 4, a method of synchronizing a fabrication schedule and a plurality of supplier schedules, according to the present invention, is illustrated. Steps include: obtaining a fabrication schedule from a fabricator data processing system (Block 100); obtaining supplier schedules from respective supplier data processing systems (Block 102); establishing restrictive links among a fabrication schedule and supplier schedules (Block 104); obtaining a change in a fabrication schedule work stage (Block 106); automatically modifying restrictive links in response to an

obtained change (Block 108); and communicating a modified fabrication or supplier schedule to a fabricator data processing system or to a supplier data processing system (Block 110).

The illustrated method is preferably recursive such that it calls itself for the purpose of determining whether changes are necessary to other schedules of network members when a given schedule has a changed time. When the starting time of a restrictively linked work stage is changed, and nothing prevents the work stage on the other end of the restrictive link from changing its starting or ending time, the Distributed Scheduling Subsystem 28 automatically modifies the restrictive link in response to the starting time change. In the illustrated embodiment, the Distributed Scheduling Subsystem 28 handles the process of checking whether work stages restrictively linked can be rescheduled. Each restrictive link contains information about the range of time within which a work stage starting or ending time can be changed. If a work stage cannot be automatically rescheduled, the changes are not made and a conflict error is returned to network members affected by the conflict.

If scheduling conflicts cannot be resolved by rescheduling, the present invention is designed to attempt to reschedule with another network member from a predetermined list of alternates. For example, if a home builder's fabrication schedule has a work stage for framing with a starting time that has slipped such that a restrictively linked framing crew A cannot perform the work, the present invention will attempt to establish a contract between the home builder and another framing crew on that home builder's list of approved framing crews. The present invention may attempt to resolve scheduling conflicts by delaying the starting times of restrictively linked schedules. If this is unsuccessful, alternative resolutions may be pursued.

FIGS. 5A, 5B, 5C are flow diagrams, illustrating a recursive method of synchronizing a fabrication schedule and a plurality of supplier schedules, according to one embodiment of the present invention, when a change is made to the fabrication schedule. To make modifications to an existing fabrication schedule, a network member (fabricator) "checks out" or obtains a copy of the fabrication schedule from the Distributed Scheduling Subsystem 28, makes the changes to the fabrication schedule, and attempts to synchronize the modified fabrication schedule with other supplier schedules. Referring now to FIG. 5A, for each fabrication schedule stored (or to be stored) within the Distributed Scheduling Subsystem 28, a CPM calculation method is performed to generate starting and ending dates of all work stages within the fabrication schedule (Block 122). This may be performed either locally via a Calendar-Driven Desktop System 60, or centrally via a Network Service Distribution System 20. For existing fabrication schedules that are being checked into the Distributed Scheduling Subsystem 28, a copy of the fabrication schedule, as it existed when it was checked out from the Distributed Scheduling Subsystem, is obtained (Block 124).

When a modified or new fabrication schedule is checked-in with the Distributed Scheduling Subsystem 28, the pre-existing starting and ending times of all work stages are compared with those of the fabrication schedule to identify any changes. This is illustrated as an iterative loop (Block 126–Block 130) in FIG. 5A. If any work stage starting times are changed, several steps are then performed. First, a work stage having a changed starting or ending time is checked to see if a lower level fabrication schedule is contained within the work stage with the changed starting or ending times (Block 132). If there is a lower level fabrication schedule

associated with a work stage having changed starting or ending times, then the lower-level fabrication schedule is recursively checked-out (Block 134). If the overall starting time (i.e., the starting time of the first work stage in the schedule) of the lower-level fabrication schedule has been changed, a CPM calculation is performed to generate new starting and ending times for each work stage (Block 136) within the lower-level fabrication schedule. The lower-level fabrication schedule is then recursively checked back in (Block 138). The recursive check-in allows for a continuous comparison of work stage starting times with original starting times. Next, a determination is made whether the lower-level fabrication schedule has been successfully checked in without any errors (Block 140). If the answer is yes, the procedure proceeds to Block 142, otherwise, the procedure proceeds to Block 164.

Referring now to FIG. 5B, after lower-level fabrication schedules have been modified and recursively checked, where necessary, a fabrication schedule work stage having a changed starting or ending time is checked for a restrictive link with a supplier schedule (Block 142). If no restrictive link exists for the changed work stage, the procedure continues in an iterative loop to the next work stage having a changed starting or ending time and checks for any restrictive links thereto (Block 144). If no restrictive links are found for any work stages having changed starting or ending times, the procedure terminates, and the fabrication schedule is checked in. When no restrictive links exist between work stages of a fabrication schedule that have been changed and supplier schedules, any changes made to the fabrication schedule have no impact on the supplier schedules.

If a fabrication schedule work stage with a changed starting or ending date is restrictively linked to a supplier schedule, this supplier schedule is located (Block 146). A determination is made whether the restrictively linked supplier schedule can be changed in accordance with the change to the work stage at the other end of the restrictive link (Block 148). If the supplier schedule can be rescheduled, the supplier schedule is checked-out by the Distributed Scheduling Subsystem 28 (Block 152), and the supplier schedule is changed accordingly (Block 154). If the supplier schedule contains work stages sequentially arranged, a CPM calculation is performed to generate starting and ending times (Block 156). The supplier schedule is then recursively checked-in with the Distributed Scheduling Subsystem 28 (Block 158). If check-in is successful (Block 160), the recursive procedure continues to the next fabrication schedule work stage with a changed starting or ending time (Block 162) and the above-described procedure is repeated. If check-in is not successful, the procedure proceeds to Block 164.

During the recursive procedure described above, if any schedule changes are not allowed, a conflict error message is returned to the appropriate subsystem within the affected network member's Calendar-Driven Desktop System 60 and no schedule changes are made. Database updates for schedule check-ins are delayed until the recursive procedure is complete. When the procedure is complete, all stored schedules are updated. If an unsuccessful check-in of a supplier schedule has occurred (Block 160) or if a restrictive link does not allow a supplier schedule to change (Block 148), the supplier schedule is marked within a temporary table as checked-in (Block 164) and an error message is returned (Block 166).

Referring now to FIG. 5C, when the iterative loop defined by Block 126-Block 130 is complete, the fabrication schedule is stored within a temporary table (Block 168). A

determination is made whether this is the first recursive call to check-in (Block 170). If the answer is yes, the temporary table is stored within the fabrication schedule database and all schedules are indicated as checked-in (Block 172). If the answer is no, a message of successful check-in is returned (Block 174).

The above-described recursive procedure is applicable to all schedules, including fabrication and supplier, that are changed and which are restrictively linked with other schedules. The present invention preferably includes additional features such as when a threshold amount of computer storage space is exceeded by changed schedules. If such a threshold is exceeded, an error message is produced. Additionally, if an attempt is made to check-out a schedule that is already waiting to be updated by the same recursive procedure, an error message is returned. If an attempt is made to check-out a schedule that is waiting to be updated by another recursive procedure, the system waits until that recursive procedure is completed.

It will be understood by those having skill in the art that one or more of the steps set forth in the flow charts of FIGS. 4 and 5A, 5B, 5C may be implemented using computer readable program code, embodied within computer usable media, executing on a general purpose computing system, on a special purpose computing system, or on a combination thereof. It will also be understood that, for the flow charts set forth in FIGS. 4 and 5A, 5B, 5C, each block, and combinations thereof, may be implemented by computer program instructions. These computer program instructions may be loaded into a computer or other programmable apparatus to produce a machine, such that the instructions which execute on the computer or other programmable apparatus create means for implementing the functions specified in the flow chart block or blocks.

The computer program instructions may also be stored in computer readable media (including magnetic media, optical media, read only memory, random access memory, and the like) that can direct a computer or other programmable apparatus to function in a particular manner, such that the instructions stored in the computer readable media produce an article of manufacture including instruction means which implement the function specified in a flow chart block or blocks. The computer program instructions may also be loaded into a computer or other programmable apparatus to cause a series of operational steps to be performed on the computer or other programmable apparatus to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide steps for implementing the functions specified in a flow chart block or blocks.

According to another aspect of the illustrated embodiment in FIG. 1, the Integrated Project Management Subsystem 66 may be constantly updated with work progress by network members. The term "work progress" includes expected work stage completion times and actual work stage completion times. When work progress shows that a work stage ending time is expected to be delayed, the Integrated Project Management Subsystem 66 may check-out the fabrication schedule, update work stage starting and/or ending times accordingly, and perform a CPM calculation to generate starting and ending times for all work stages in the fabrication schedule. The Integrated Project Management Subsystem 66 may then attempt to check-in the fabrication schedule into the Distributed Scheduling Subsystem 28. Depending on the existence of restrictive links, this operation may succeed or fail. If it fails, the Integrated Project Management Subsystem 66 may receive a list of conflicting

restrictive links from the Distributed Scheduling Subsystem 28 which are used to develop solutions to the scheduling conflict(s). Depending on the nature of a particular project or work stage, the Integrated Project Management Subsystem 66 may cancel the originally scheduled process or relationship with a supplier and establish a new one with another network member supplier. The Integrated Project Management Subsystem 66 may also force a delay on a work stage starting time, re-compute the fabrication schedule, and attempt to check-in the fabrication schedule with the Distributed Scheduling Subsystem 28. According to one aspect of the present invention, the Integrated Project Manager Subsystem 66 may be designed to automatically track fabrication and supplier schedules of network members and keep them synchronized or find solutions when scheduling conflicts arise.

Home Building Industry Example 1

In the home building industry, a home builder typically has several projects (houses) under construction and others that are about to start. Each project has a fabrication schedule with multiple, sequentially arranged work stages stored in a Distributed Scheduling Subsystem 28. As the home builder enters into contracts for materials and labor for each work stage, restrictive links are established among the fabrication schedule and the multiple supplier schedules. The process of forming contracts with other network member suppliers is preferably performed via the General Transaction Services Subsystem 32. When a contract is formed between the home builder and a supplier, the Integrated Project Management Subsystem 66 "checks-out" the home builder's fabrication schedule for this project. Preferably, at about the same time, the Integrated Work Activity Calendar 64 for the home builder checks-out the activity calendar for the subcontractor/supplier. Both systems (the Integrated Project Management Subsystem 66 and Integrated Work Activity Calendar 64) indicate that a restrictive link has been established between at least one work stage in the fabrication schedule and a supplier schedule, per the established contract. Both systems (the Integrated Project Management Subsystem 66 and Integrated Work Activity Calendar 64) check-in the restrictively linked fabrication and supplier schedules. Preferably, the Distributed Scheduling Subsystem 28 recognizes restrictive link indicators and establishes actual restrictive links between the respective schedules.

Preferably, work progress is recorded into the Integrated Project Management Subsystem 66 which calculates the starting and ending times of the fabrication schedule work stages using CPM calculation methods. The fabrication schedule is checked-in and a changed starting time for a work stage is identified. The home builder's Work Activity Calendar is contacted to see if the time change can be accommodated. If yes, the starting time change is made. If no, the check-in fails and an error message is returned to the Integrated Project Management Subsystem 66 of the home builder.

Assigning Float Time to Work Stages Within a Critical Path Schedule

A critical path schedule includes a sequential arrangement of work stages, some of which cannot be performed until a prior one is performed. If a starting time is known for a given schedule, the starting and ending times for each work stage in the schedule can be computed utilizing various CPM calculation methods known to those having skill in the art.

A delay to one work stage in a critical path will have a cascade effect on the schedule such that for each day a work stage is delayed, a delay to the entire schedule results. Work stages that are not on the critical path are considered to have float. A work stage with three days float can exceed its scheduled duration by three days without affecting the overall schedule.

Referring now to FIGS. 2A, 2B, 2C, four different projects 70a, 70b, 70c, 70d, each with three work stages slated to last one day, are illustrated. Alone, each project should take three days to complete; however, some of the work stages must await the completion of work stages in other projects. With such an interdependency of work stages, a single change to a work stage can cause a large sequence of cascade schedule changes. FIG. 2B clarifies the actual flow of completion for each project illustrated in FIG. 2A. Projects 1, 3 and 4 can begin on day one, but Project 2 is delayed by a day until the completion of work stage 1 in Project 1. Though Project 3 can start on day one, it has a two day delay before work can continue as it awaits the completion of work stage 2 for Project 2, and so forth. Though requiring the same number of days to complete, and even though it began at the same time as Project 1, Project 4 requires three extra days to complete because of work stage 3 of Project 3.

FIG. 2C illustrates the cascade effect caused to the four projects in FIGS. 2A and 2B when a single work stage in one of the projects is delayed. Because work stage 3 in Projects 1 and 2, and work stage 2 in Project 3 are all dependent upon the completion of work stage 2 in Project 2, they are each delayed until work stage 2 in Project 2 is completed.

According to another aspect of the present invention, the Distributed Scheduling Subsystem 28 prevents small schedule changes to critical path schedules from causing large cascading schedule changes. This is accomplished by adding a quantity of time, referred to as "float time", to the starting time of a selected work stage or activity, as illustrated in FIGS. 3A and 3B. FIG. 3A illustrates multiple work stages 84-90 which define a critical path. Work stages 85, 88 include preceding float time 85a, 88a, respectively, prior to their starting time. As a result of the preceding float time, work stages 85, 88 are not on the critical path that determines the minimum length of the schedule. Note that there is a two day difference between the start of work stage 88 with preceding float 88a and work stage 87.

FIG. 3B depicts the effects on the schedule illustrated in FIG. 3A as a result of a one day delay to the first work stage 84. Consistent with CPM scheduling techniques, each work stage on the critical path is delayed by one day, causing the entire schedule to take one day longer than originally planned. However, the actual work days scheduled for work stage 85 remained the same and the available preceding float 85a has simply been reduced. The entire cascade potential on the subsequent work stage with preceding float time has thus been absorbed. As a result, the preceding float time remains unchanged for work stage 88 even though other work stages around it, have been adjusted. Note the two day gap between work stages 87 and preceding float 88a (FIG. 3A) has been reduced to only 1 day. Utilizing the preceding float allowed the fabricator's system to manage the start times after the delay so that the system did not automatically bump the succeeding work stage back an additional day. Rather, the system utilized the pre-determined preceding float in order to hold the later work stages to their original start dates, therefore not all work stages were changed, thus not all suppliers were affected.

According to the present invention, if a work stage in the critical path has an ending time later than scheduled, pre-

ceding float time on following work stages can prevent them from being rescheduled. When a work stage starting or ending time is modified, the preceding float time is adjusted in order to keep the actual work stage starting time the same. Preferably, the preceding float quantity is restricted such that it cannot go below zero or above a maximum preceding float quantity.

Automatic Product Information Updating

Typically, product distributors purchase products from product manufacturers and sell them to retailers and end users. When product manufacturers make changes to their products, or change information about their products, distributors typically accumulate this information and retransmit it (in printed or electronic form) to retailers and end users. The task of collecting and re-transmitting product information can be daunting. The present invention, through the Network Service Distribution System 20, makes product manufacturer information available to all network members who need the information without the duplicative efforts of distributors. The present invention allows product information to flow directly from manufacturers to retailers, distributors, and end users without interfering with the established channels of trade and business relationships.

The present invention, via the Distributed Product Data Management Subsystem 34, facilitates the electronic distribution of product information from network members who are product manufacturers to other network members. Because network members can view product information electronically and because the information is updated by manufacturers and automatically distributed down through the network, network members who are distributors do not have to maintain and redistribute constantly changing manufacturer product information.

According to one aspect of the present invention, a computer based product catalog system is provided. The computer based product catalog system includes multiple electronically stored manufacturer catalogs which include descriptive text, numerical data (measurements), multimedia information (video, audio, etc.) describing items manufactured by respective manufacturers, and based on standardized data models for different product types. An electronically stored distributor catalog including links to manufacturer product information, modifications to the descriptions of selected items from the various manufacturers, and distributor specific data regarding availability, deliver schedules, and the like. Also included within the system is a subsystem for automatically updating the descriptions of items within the manufacturer catalogs, and a subsystem, responsive to the automatic updating subsystem, for maintaining the links to items within the distributor catalog. A terminal, operatively connected to the manufacturer catalogs and to the distributor catalog, may be provided for viewing item descriptions within the manufacturer and distributor catalogs at the retail level of product distribution. A subsystem may be provided within the terminal for ordering items from the manufacturer and distributor catalogs.

According to an embodiment of the present invention, each manufacturer may access the Distributed Product Data Management Subsystem 34 to enter and maintain product information, including images, specifications, descriptions, and wholesale and/or retail pricing information. Each distributor may access the Distributed Product Data Management Subsystem 34 to select and maintain the list of products they will provide from any number of different

manufacturers. Retailers, end users, and the like, can view the products offered by all network member distributors, including the manufacturer's product information, without concern that the information is outdated or inaccurate. The present invention facilitates the organization and presentation of product information in numerous formats. For example, all products of a given type, all products by manufacturer, may be grouped and viewed together. Network members may also view product information directly from a given manufacturer.

Referring now to FIG. 6, the distribution of product information, according to one aspect of the present invention, is schematically illustrated. A variety of unique on-line product catalogs are shown being generated and updated without the need for distributors to reorganize and retransmit them as manufacturer specifications change. As shown, all products originate from manufacturers, (whether a single artisan fabricating unique works of art, or a major company creating thousands of reproductions of a single product). In all cases, a digital catalog record of textual descriptions, specifications, drawings, photographs, video clips, animation, price information, and any other desired information, is created for each product and made available to network members via the Network Service Distribution System 20.

In the illustrated embodiment shown in FIG. 6, Manufacturer 1 maintains an electronically stored catalog 200 of products for sale directly to distributors, retailers, and end users, each of whom is a network member. Each is allowed to view all or portions of Manufacturer 1's catalog 200, according to Manufacturer 1's discretion. End user 1 (208) may view the catalog 200 of Manufacturer 1 and may view the regional catalog 204 of Distributor 1 as indicated by single-arrowhead connectors 232. End User 1 (208) may purchase products from Retailer 1 (206) or from Manufacturer 1, as indicated by double-arrowhead connectors 234. Distributor 1 maintains a catalog 202 of products purchased from Manufacturers 1, 2, and 3 via their respective catalogs 200, 210, and 212, as indicated by double-arrowhead connectors 234.

Distributor 1 markets and distributes selected products from the catalogs of Manufacturers 1, 2, and 3 (200, 210, 212). The result is a unique catalog having a variety of products. Should any manufacturer update its product information via the Network Service Distribution System 20, it is automatically updated within Distributor 1's general catalog 202 and regional catalog 204. Because Distributor 1 allows Retailer 1 (206) and the End User 1 (208) to view its regional catalog 204, the updated product information from Manufacturer 1 is available without Distributor 1 having to reformat or retransmit the product information.

Still referring to FIG. 6, Manufacturer 2 supplies products to Distributor 1 and 2. Distributor 1 and 2 include information about these products within their respective catalogs 202, 214. This results in unique catalogs of products available to Retailer 1 (208) and Retailer 2 (218). In the illustrated embodiment, Retailer 1 (208) may view Manufacturer 2's catalog 210, as indicated by single-arrowhead connector 232, even though purchasing of products is only available through Distributor 1. Retailer 2 (218) may view Manufacturer 2's catalog 210, but can purchase only from Distributor 2 via Distributor 2's regional catalog 216. Manufacturer 3 is different from Manufacturer 2 in that only Distributors 1 and 2 may view or purchase products from its catalog 212, as indicated by double-arrowhead connectors 234. In the illustrated embodiment, a Specialized Provider has a unique catalog 220 of products purchased from Retailer 1 and 2

(206, 218) as indicated by double-arrowhead connectors 236. End User 1 (208) may view and purchase products from the Specialized Provider catalog 220 as indicated by double-arrowhead connector 238.

As illustrated in FIG. 6, the present invention facilitates the creation and automatic maintenance of many electronically stored product catalogs. In the illustrated embodiment, thirteen different catalog configurations containing products created by 3 different manufacturers are available to network members. Rather than thirteen different catalogs requiring reprinting or updating every time a manufacturer makes a change to specifications or product information, the present invention provides real-time updates as soon as they are made by a manufacturer.

Both the Network Service Distribution System 20 and each Calendar-Driven Desktop System 60, according to the present invention, may be implemented via a variety of computing devices, including, but not limited to, mainframe computing systems, mini-computers, and personal computers. It will be understood that a computer or other apparatus configured to execute the program code, embodied within computer usable media, operates as means for performing the various functions and carries out the methods of the various operations, according to the present invention. Stored computer readable program code also acts as a means for carrying out the various methods and functions of the present invention.

Computer readable program code means is provided for retrieving a copy of a first schedule from the network service distribution system, for changing at least one work stage starting or ending time in the copy of the first schedule, and for automatically changing the starting and ending times of work stages in the stored first schedule and a second stored schedule restrictively linked to the first schedule as a result of changes made to the copy of the first schedule. In particular, computer readable program code means is provided for each of the following: comparing a copy of a first schedule with the first schedule stored within the network service distribution system to identify changed work stage starting and ending times; identifying whether a restrictive link exists between a work stage of a first schedule and a work stage of a second schedule; determining whether the identified restrictive link permits the starting or ending times of the second schedule to be changed; and automatically changing the starting and ending times of the second schedule work stage, thereby synchronizing the first and second stored schedules.

The present invention may be written in various computer languages including, but not limited to, C++, Smalltalk, Java, and other conventional programming languages such as BASIC, FORTRAN and COBOL. The Network Service Distribution System 20 and the Calendar-Driven Desktop System 60 preferably run on current standard desktop computer operating systems such as, but not limited to, Windows®, Windows 95®, Windows NT®, UNIX®, and OS/2®. The present invention utilizes, in part, many standard features of current desktop configurations, such as the ability to store data locally, connect to the Internet, and display visual information.

The foregoing is illustrative of the present invention and is not to be construed as limiting thereof. Although a few exemplary embodiments of this invention have been

described, those skilled in the art will readily appreciate that many modifications are possible in the exemplary embodiments without materially departing from the novel teachings and advantages of this invention. Accordingly, all such modifications are intended to be included within the scope of this invention as defined in the claims. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents but also equivalent structures. Therefore, it is to be understood that the foregoing is illustrative of the present invention and is not to be construed as limited to the specific embodiments disclosed, and that modifications to the disclosed embodiments, as well as other embodiments, are intended to be included within the scope of the appended claims. The invention is defined by the following claims, with equivalents of the claims to be included therein.

What is claimed is:

1. A computer based product catalog system for distributing product information in real time to separate business entities within the building industry, comprising:

- a communications network;
- a first data processing system of a first product manufacturer within the building industry in communication with the communications network, wherein the first data processing system comprises:
 - a first product catalog that includes descriptions of first products which are manufactured by the first product manufacturer within the building industry; and
 - first means for updating the descriptions of the first products in the first product catalog;
- a second data processing system of a second product manufacturer within the building industry in communication with the communications network, wherein the second data processing system comprises:
 - a second product catalog that includes descriptions of second products which are manufactured by the second product manufacturer within the building industry; and
 - second means for updating the descriptions of the second products in the second product catalog;
- a third data processing system of a product distributor within the building industry in communication with the communications network, wherein the third data processing system comprises a third product catalog that includes selected ones of the descriptions of the first products and the second products;
- a retail terminal of a product retailer within the building industry in communication with the communications network, wherein the retail terminal comprises means for displaying product descriptions in the first, second, and third product catalogs; and
- third means, responsive to the first and second means, for automatically updating in real time the selected ones of the descriptions of the first and second products in the third product catalog.

2. A computer based product catalog system according to claim 1 wherein the retail terminal further comprises ordering means, for ordering items from the first, second, and third product catalogs, via the retail terminal.

* * * * *



US005870717A

United States Patent [19] Wiecha

[11] Patent Number: **5,870,717**
[45] Date of Patent: **Feb. 9, 1999**

[54] **SYSTEM FOR ORDERING ITEMS OVER
COMPUTER NETWORK USING AN
ELECTRONIC CATALOG**

5,315,504 5/1994 Lemble 395/650
5,319,542 6/1994 King, Jr. et al. 235/383
5,570,291 10/1996 Dudle et al. 364/188
5,576,951 11/1996 Lockwood 395/227

[75] Inventor: **Charles Francis Wiecha**, New York,
N.Y.

[73] Assignee: **International Business Machines
Corporation**, Armonk, N.Y.

Primary Examiner—Frantzy Poinvil

Attorney, Agent, or Firm—Stephen C. Kaufman, Esq. IBM
Corporation; Scully, Scott, Murphy & Presser

[21] Appl. No.: **558,065**

[22] Filed: **Nov. 13, 1995**

[51] Int. Cl.⁶ **G06F 153/00**

[52] U.S. Cl. **705/26; 235/385**

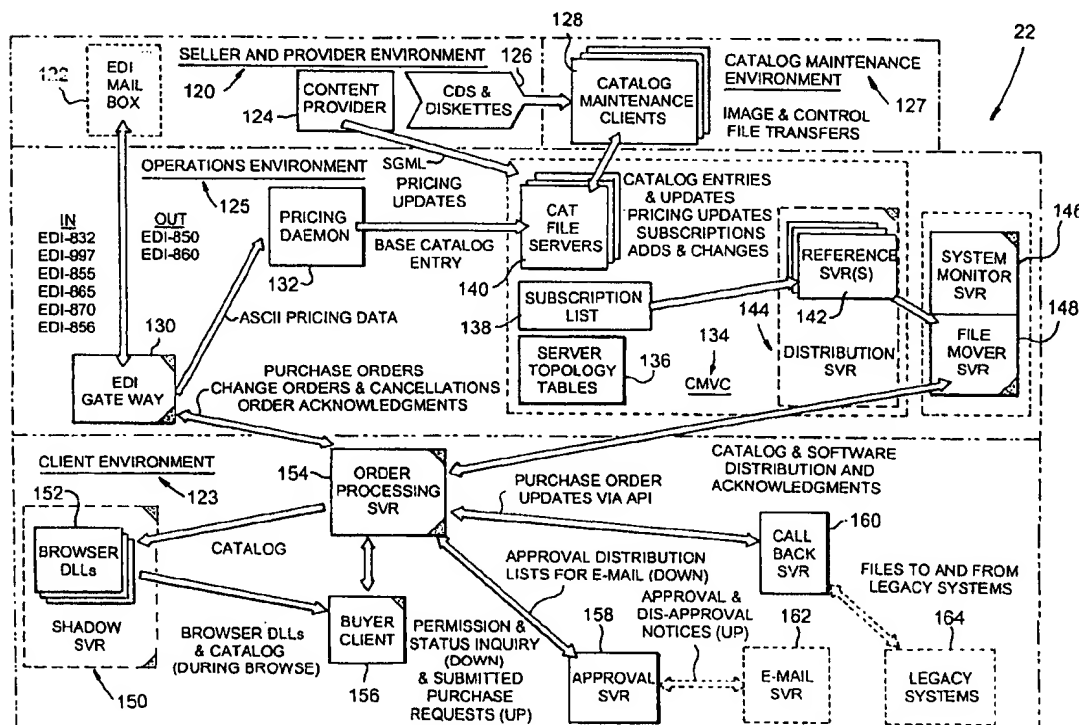
[58] Field of Search 395/201, 226-228,
395/244; 379/91.01; 340/825.26-825.28,
825.33-825.35; 283/56; 235/378-381, 385;
902/22, 30-33; 705/1, 26-28; 707/1, 10,
200

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,992,940 2/1991 Dworkin 235/383

6 Claims, 12 Drawing Sheets



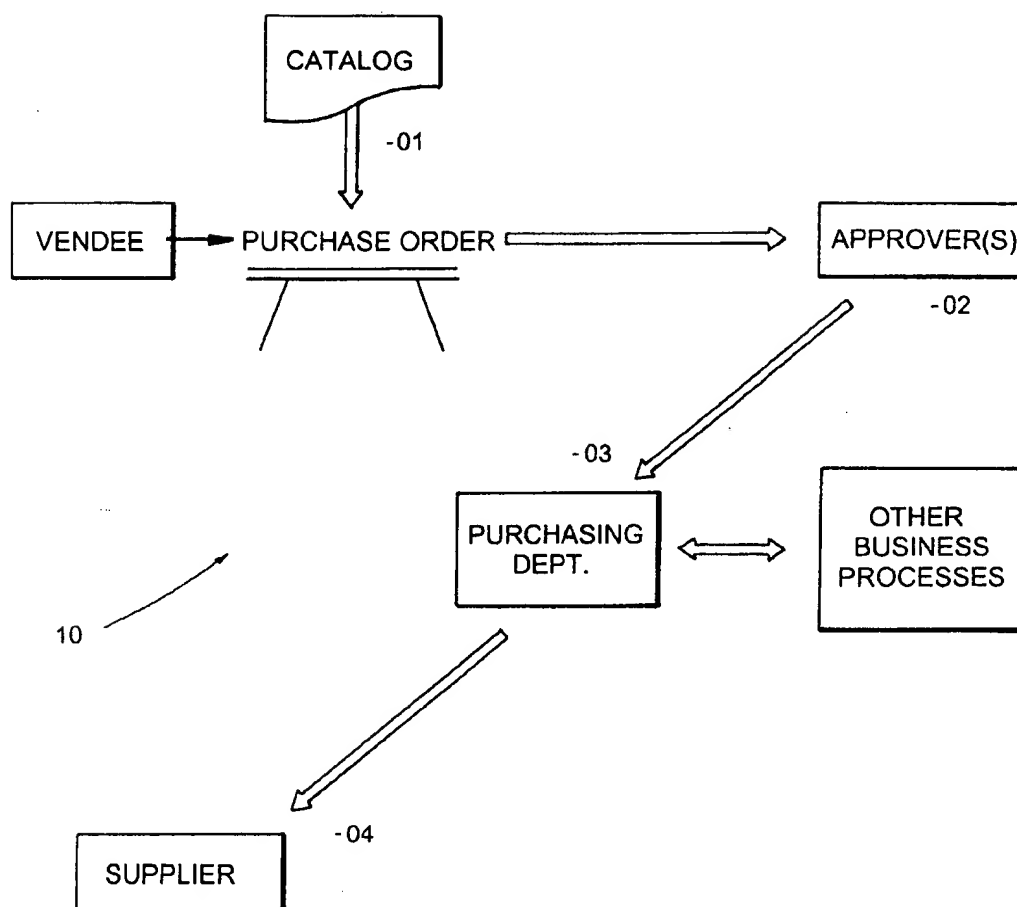


Fig. 1
(Prior Art)

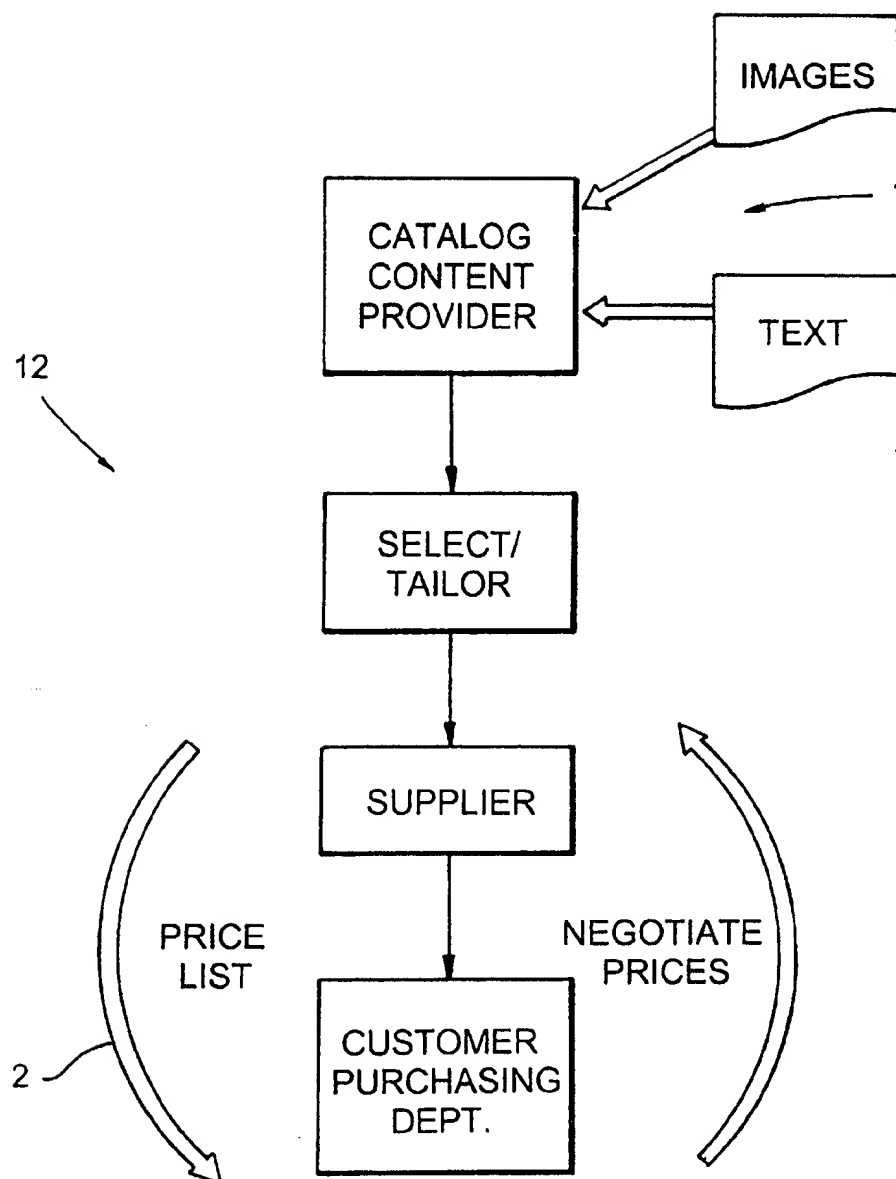


Fig. 2
(Prior Art)

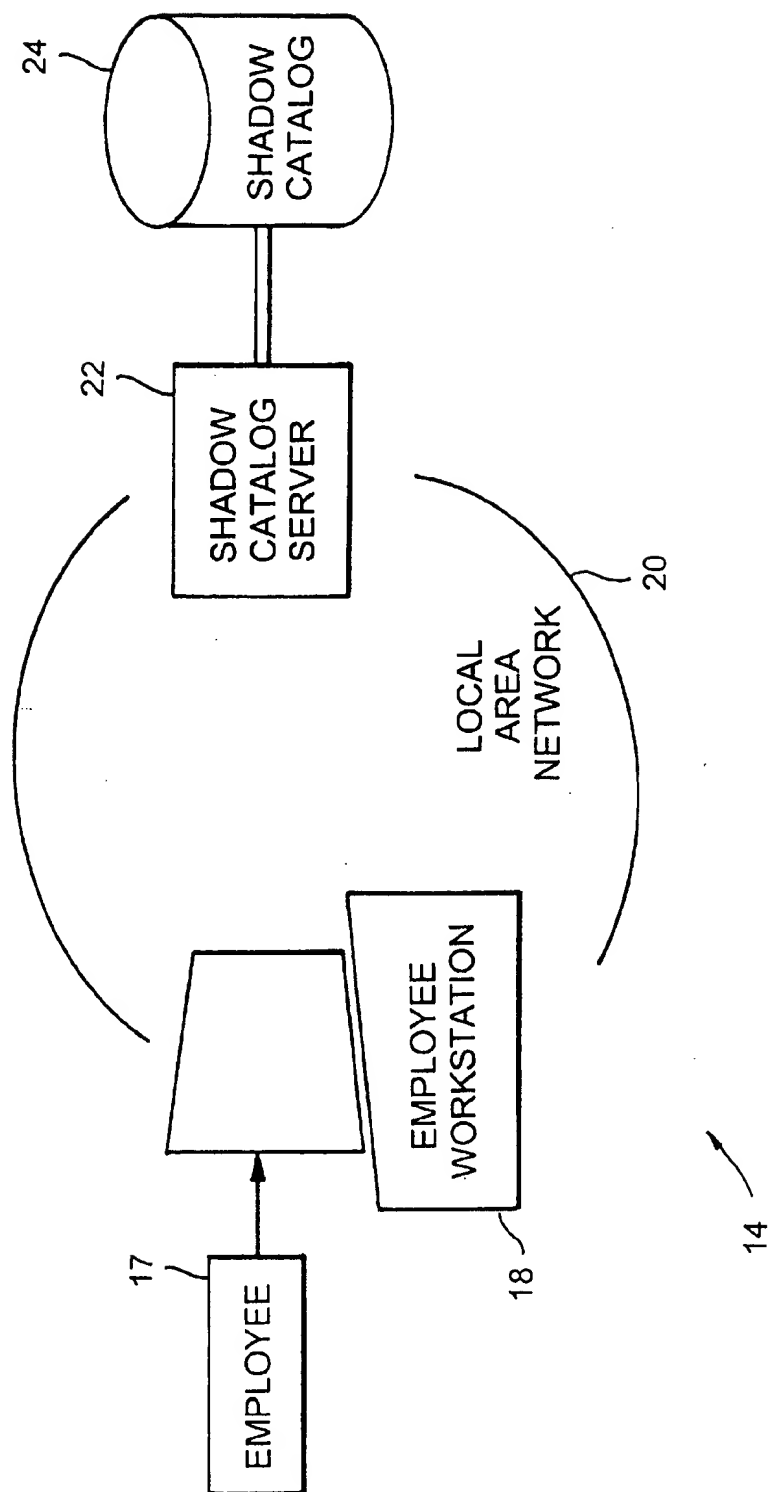
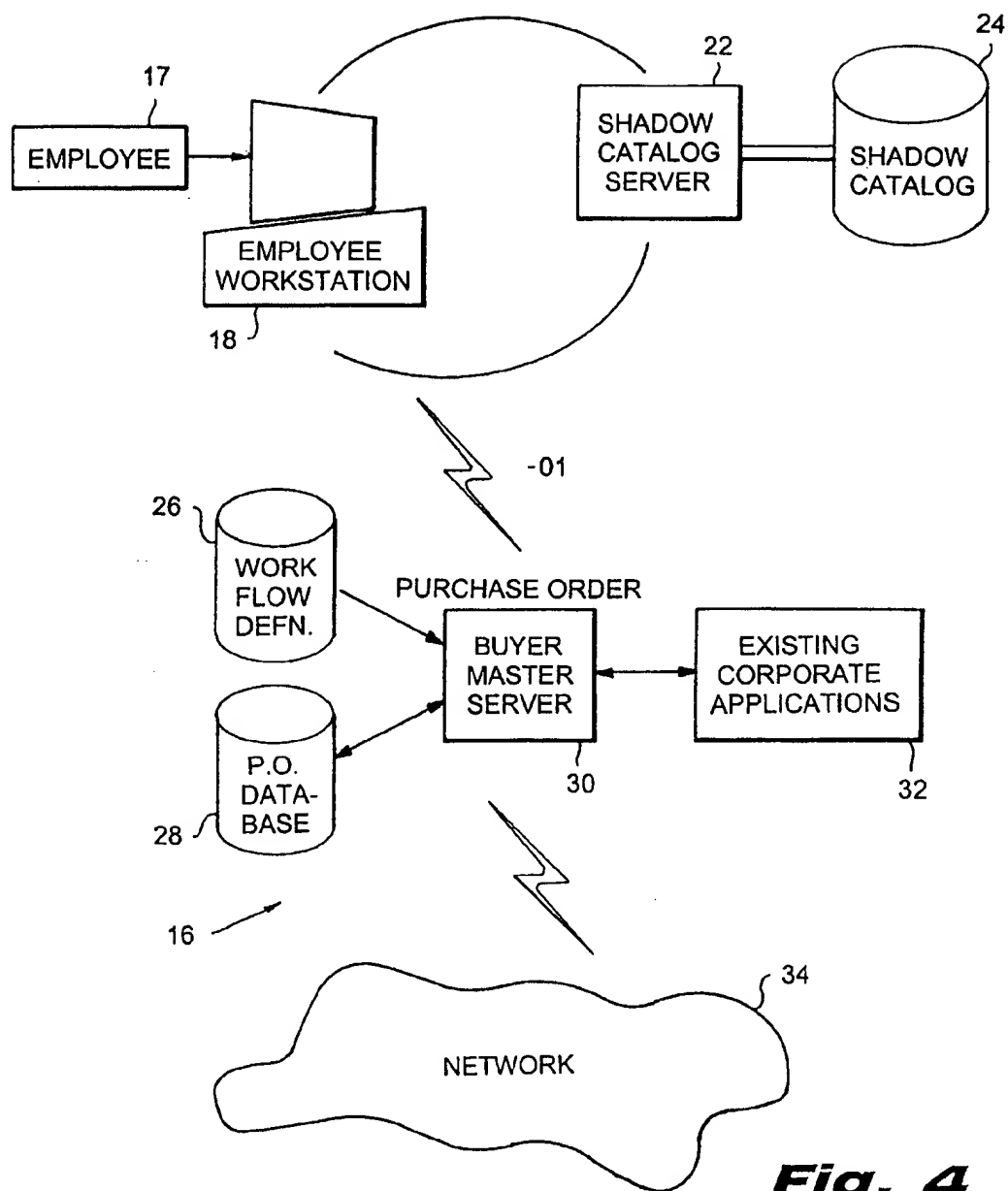


Fig. 3



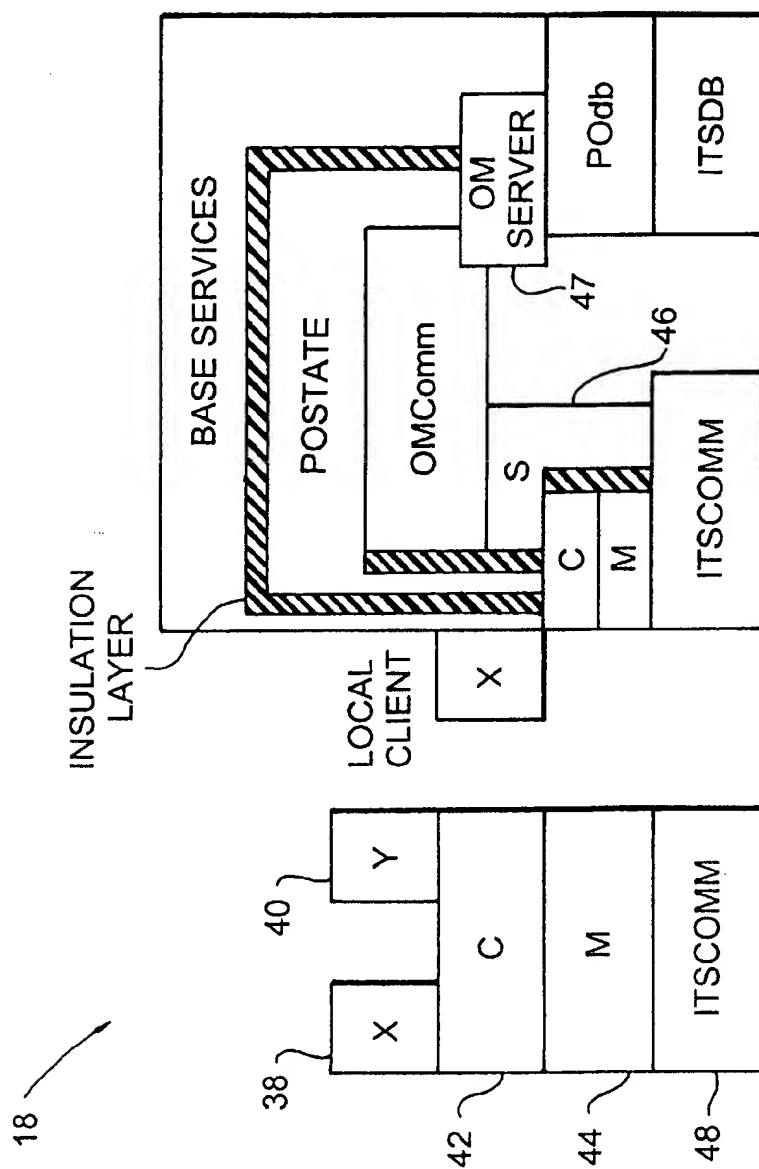
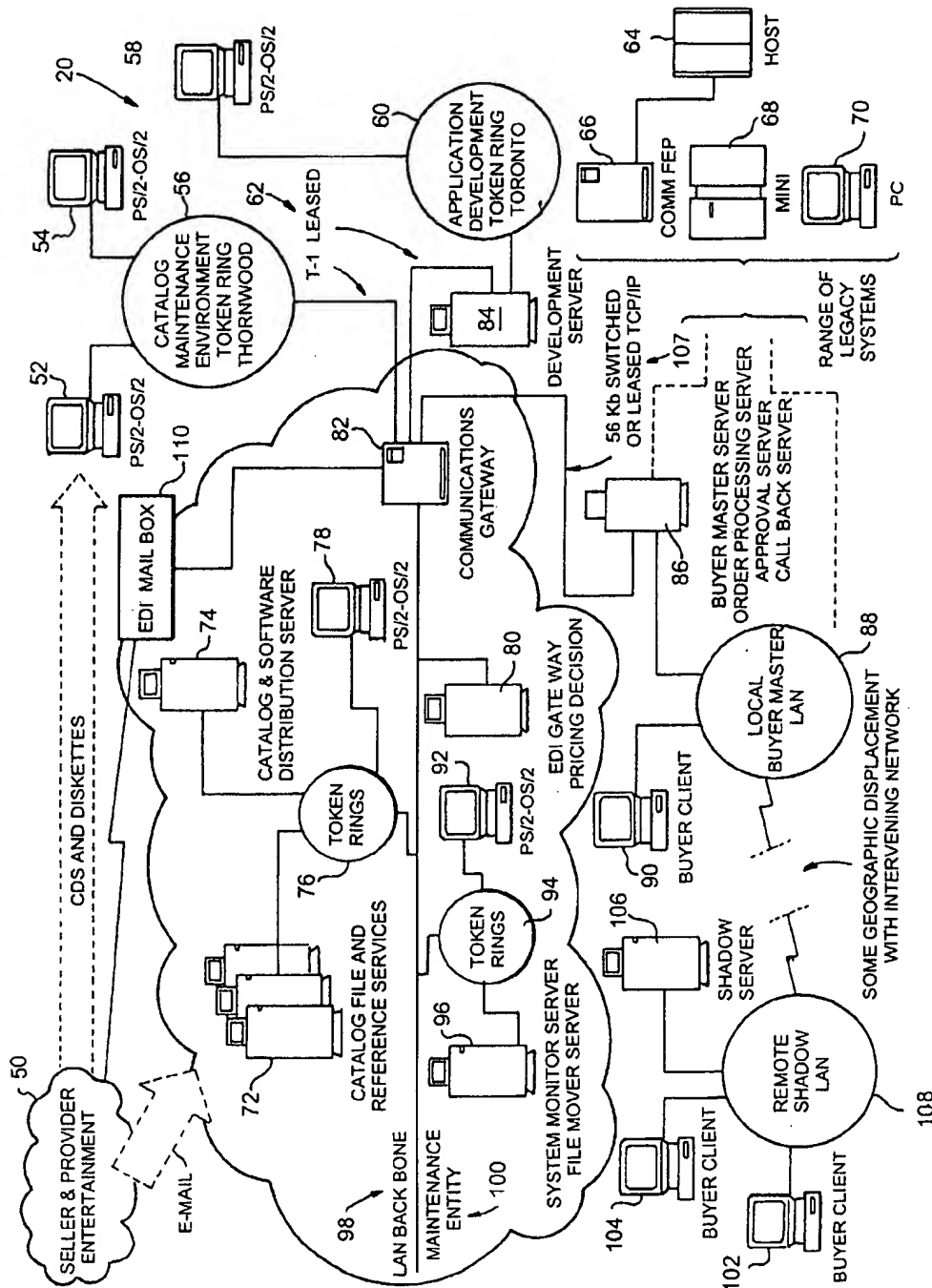


Fig. 5

**Fig. 6**

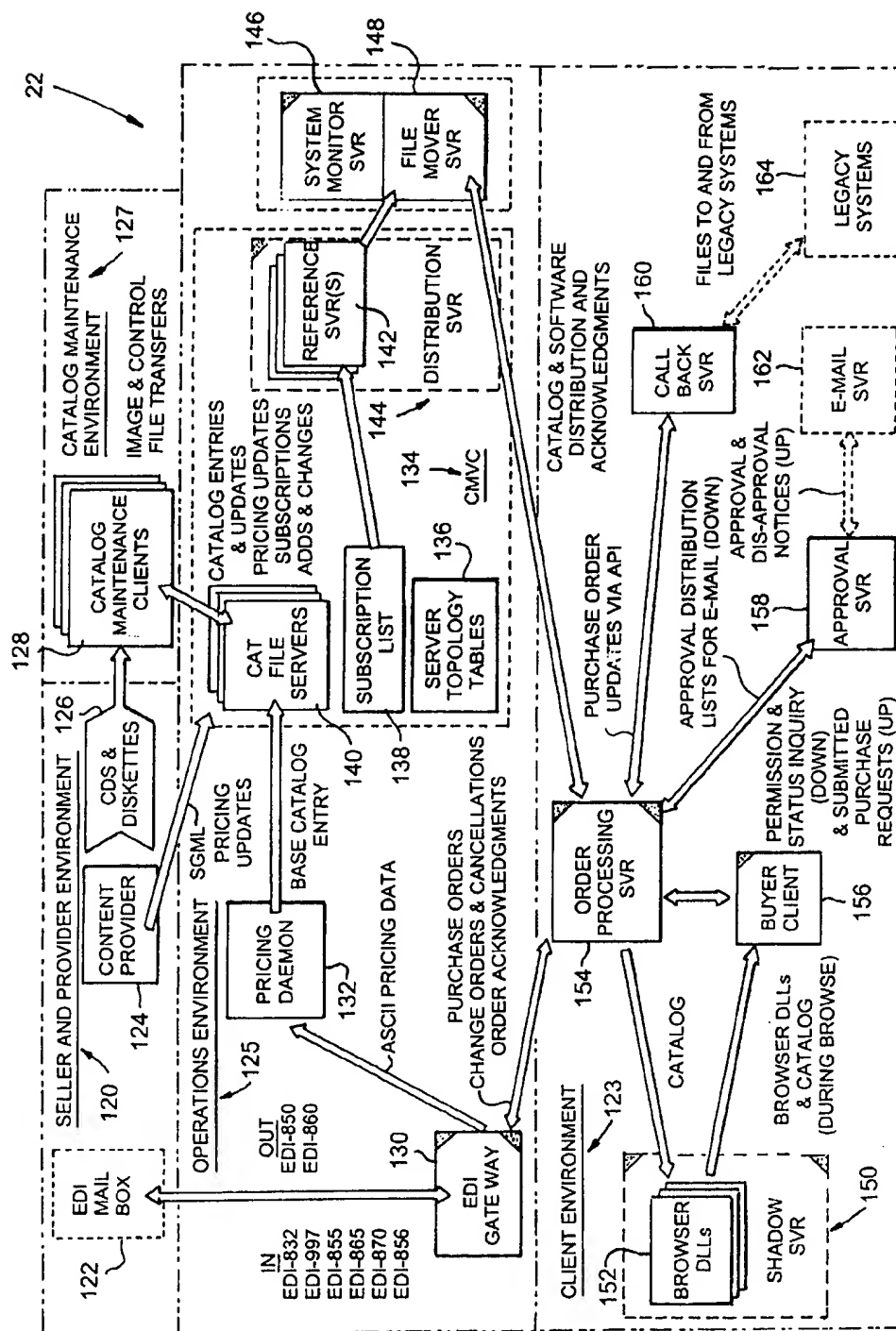
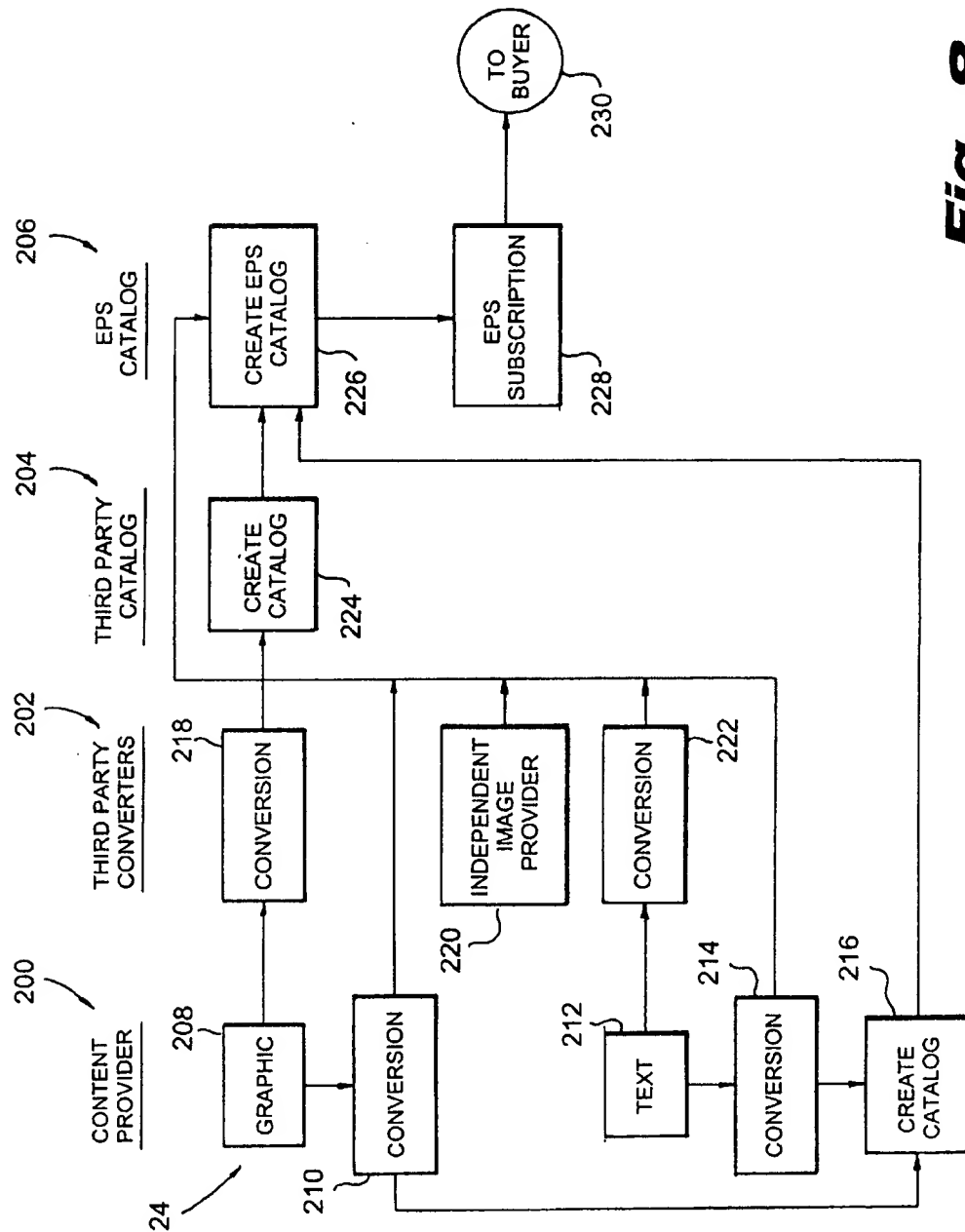


Fig. 7

**Fig. 8**

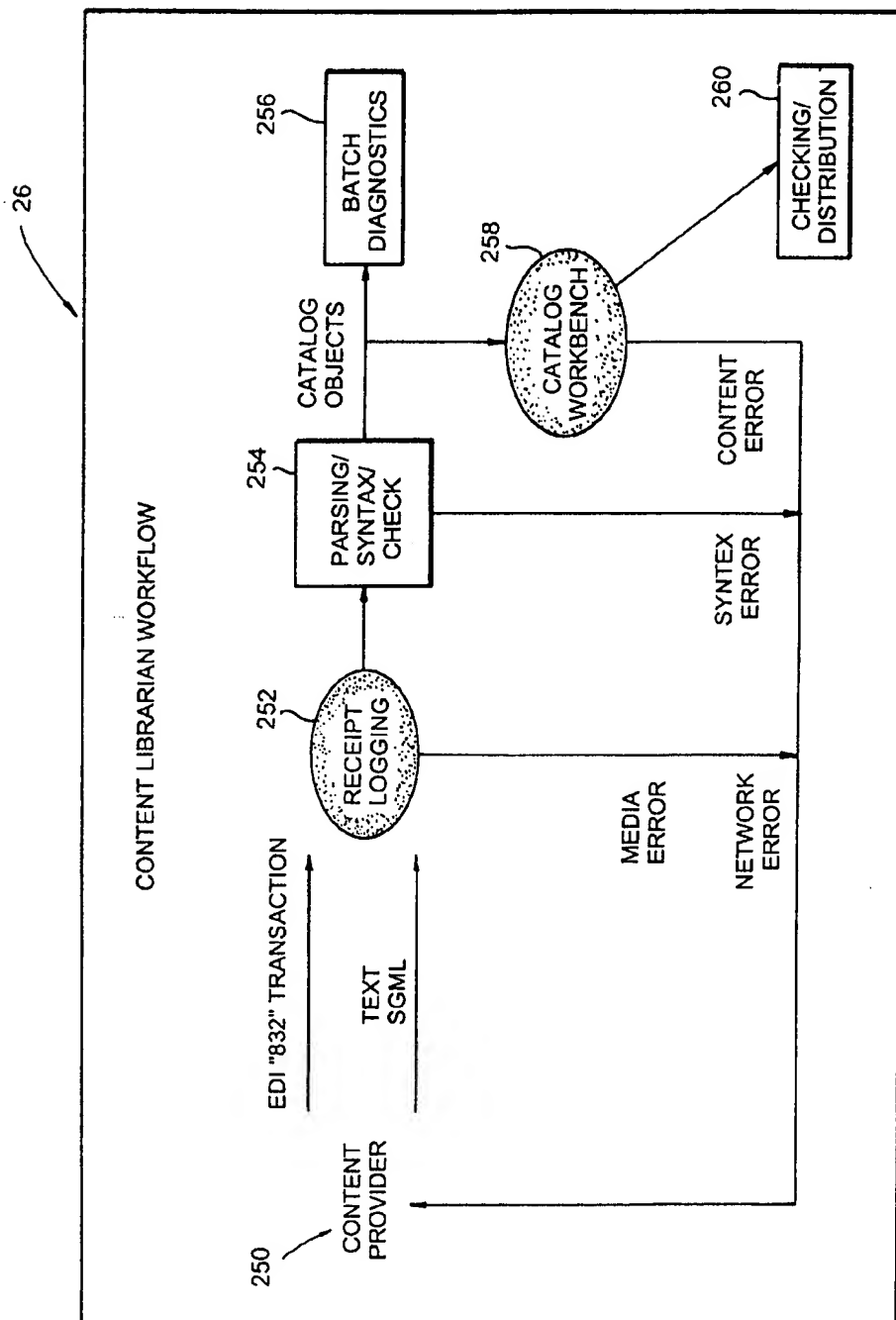


Fig. 9

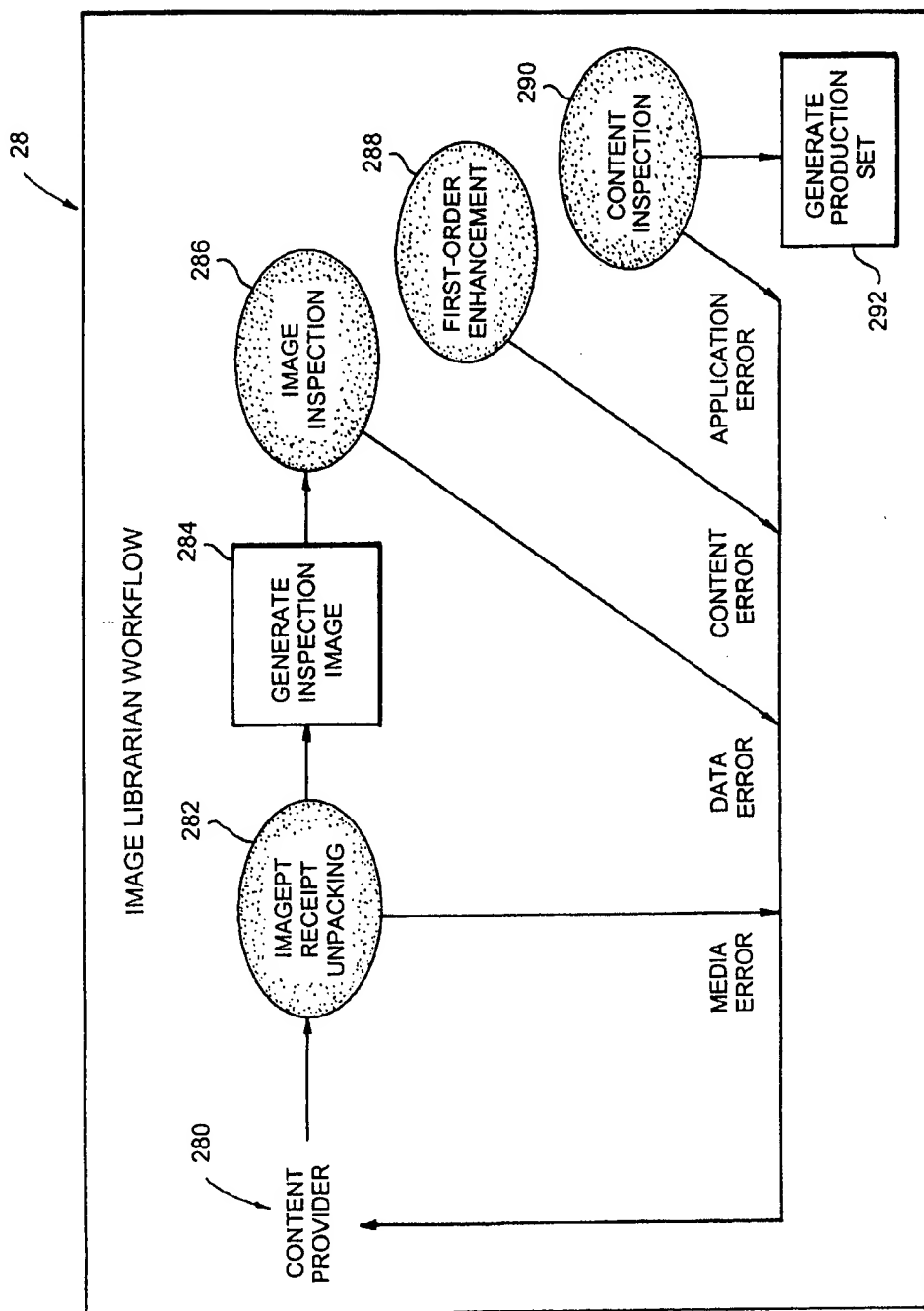


Fig. 10

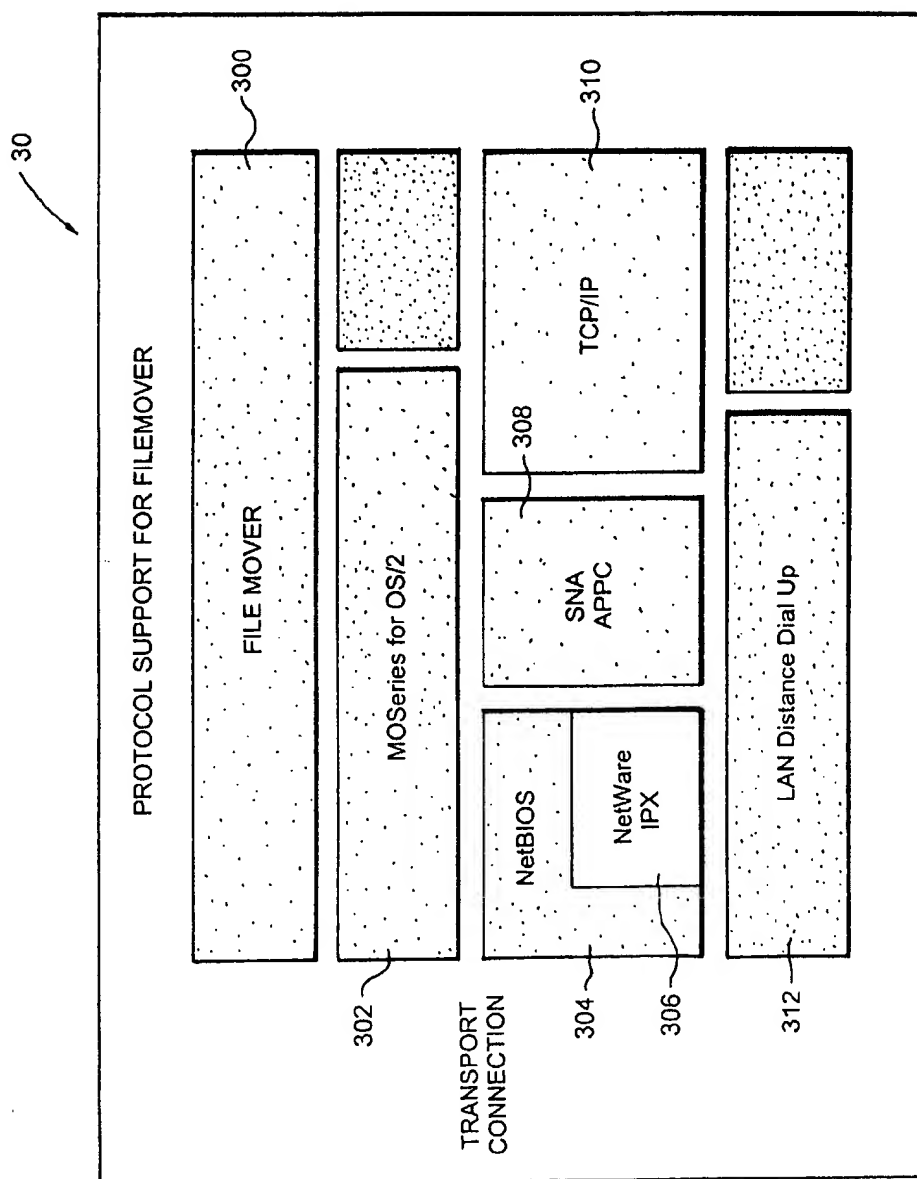
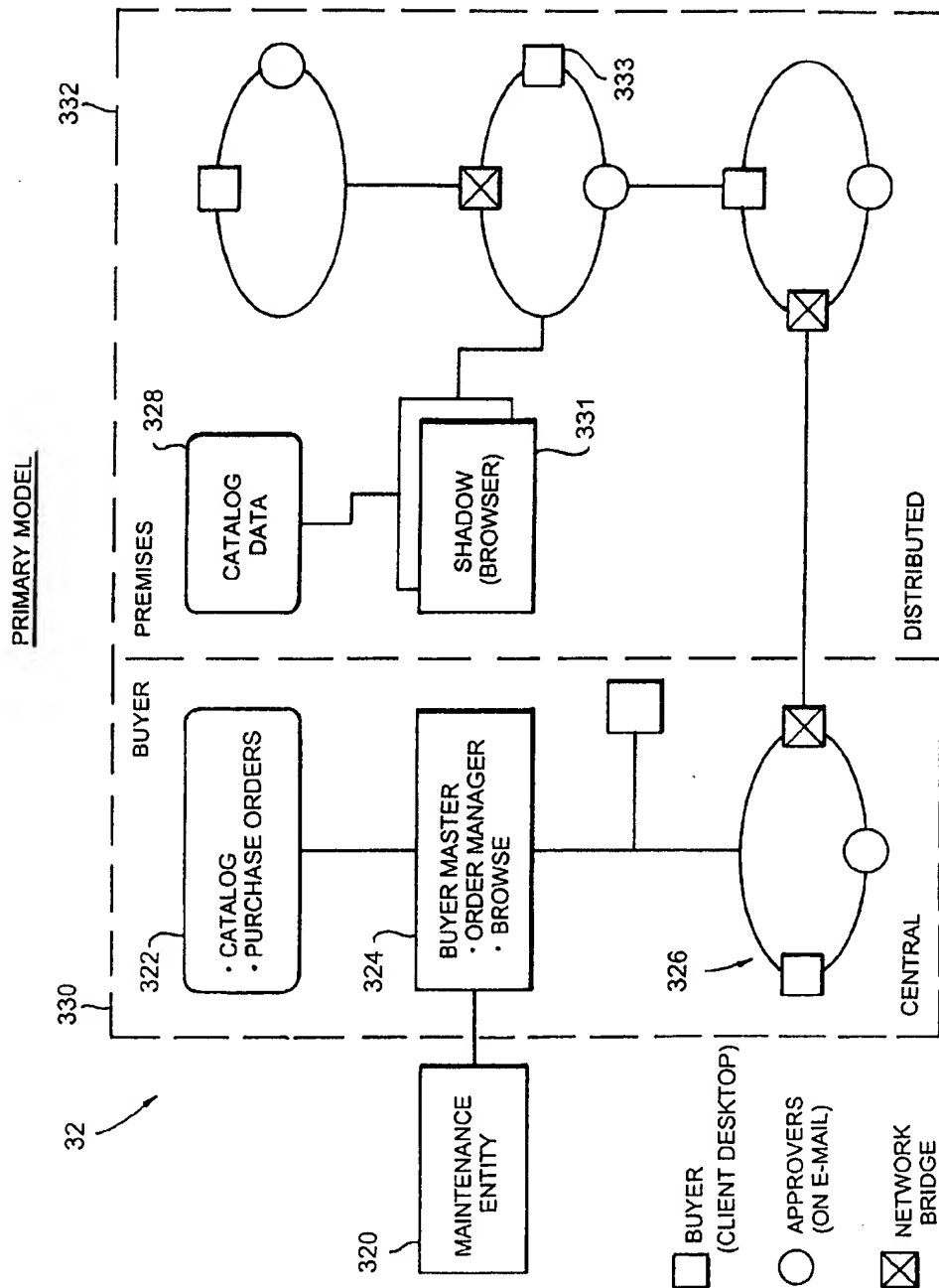


Fig. 11



SYSTEM FOR ORDERING ITEMS OVER COMPUTER NETWORK USING AN ELECTRONIC CATALOG

FIELD OF THE INVENTION

This invention relates to a system for electronically linking a buyer and a seller in a purchasing cycle.

INTRODUCTION TO THE INVENTION

Current procurement procedures in corporations of all sizes are largely manual, or at best semi-automated by electronic mail, and are as a consequence labor-intensive and costly. Typical procedures operate as described below. Refer to the attached drawings, which make the description easier to follow:

1. Typically, an employee requiring some item (such as a piece of office equipment) will look for it in a supplier's catalog. Such catalogs may be at the user's workstation, but are often kept in some central location for general reference (see FIG. 1, numeral 10).

2. The employee transcribes information (such as part numbers and price) from the catalog on to a purchase order (FIG. 1, step 01).

3. The completed purchase order then normally goes through an approval process, which may, depending on the value of the order and the nature of the items on it, require several sign-offs by people within the corporation with budgetary responsibilities (FIG. 1 step 02).

4. The purchase order is then sent to the purchasing department of the corporation, which checks the information. These checks include verifying that the items are being ordered from the correct supplier: the same item may be offered by more than one supplier, but contracts are frequently negotiated that require a particular item to be procured from only one of these suppliers, and the Purchasing department has to verify this (FIG. 1, step 03).

5. The Purchasing department then sends the Purchase Order to the supplier (FIG. 1, step 04).

The process described above can take weeks if not months.

At the other end of the procurement process, printing and distributing the traditional paper catalog is also a labor-intensive and costly process (see FIG. 2, numeral 12), especially since catalogs are usually of high quality in terms of art-work and color reproduction (FIG. 2, step 01) but may be distributed at no charge to potential buyers. The long production time for a typical catalog (of the order of several months) causes additional problems when it includes items such as personal computers and consumer electronics where prices are highly volatile. Vendors have sought a number of solutions to this problem, the most common being to omit prices from the catalog altogether and issue a separate list of prices (FIG. 2, step 2) several times throughout the lifetime of the catalog.

In response to this situation, we now disclose a novel system for ordering items. The system comprises:

- 1) means for receiving and processing images and text from a plurality of catalog content providers for creating and maintaining one or more electronic catalogs in a central location for subsequent distribution over a computer network;
- 2) means for receiving supplier's price and catalog changes and propagating them to one or more selected buyers over a computer network;

3) a first end-user computer system comprising user interface and able to access disk storage on a shadow catalog server;

4) a shadow catalog server which comprises a second computer system located within the enterprise whose disk storage can be accessed over a local area network by one or more end-users' computers in an efficient manner; said disk storage being used to hold (1) one or more electronic catalogs, and (2) program code comprising a "Catalog Browser" capable of transmitting purchase orders to a master buyer and server;

5) a master buyer server comprising a third computer system located within an enterprise containing (1) program code comprising an order manager and a purchase order workflow which takes purchase orders from one or more end-user computers and and controls their flow through the enterprise's business processes before transmitting them over a network to the supplier; and (2) a purchase order data base.

The advantages of the novel system are appreciated by contrasting it to the prior art summarized above. In particular, the concept of corporations ordering items from suppliers over a computer network is well-established, and has led to the formalization of EDI (electronic data interchange).

The concept of consumers or end-users (the people who will ultimately make use of an item) ordering items from electronic catalogs over a (usually public) network is also well-established. Public network services such as Prodigy (tm), America On Line (tm) and Compuserve (tm) allow subscribers to their services to select consumer and household items from catalogs placed there by suppliers. The items are typically shipped to the subscriber's home, and the cost charged against a credit card. Such systems have to date only allowed the subscriber to access one supplier's catalog at a time.

It may be noted that neither of the above approaches is a complete solution to the problem addressed by the disclosed invention, which is to allow end-users within a corporation to order necessary items as if they were consumers ordering items for their own use and at their own expense, but to have such orders then flow through the enterprise's normal business controls before being submitted to the supplier. The disclosed invention also goes beyond these solutions in allowing the catalog an end-user sees to be sub-setted and otherwise modified from the supplier's general catalog; in facilitating comparisons between items from several suppliers; and in permitting orders to be generated containing items from several suppliers.

BRIEF DESCRIPTION OF THE DRAWING

The invention is illustrated in the accompanying drawings which:

FIGS. 1 and 2 show prior art purchasing and catalog creation processes;

FIG. 3 shows an end-user environment;

FIG. 4 shows a client environment;

FIG. 5 shows a sample screen for features of the present invention;

FIG. 6 shows an invention topological overview;

FIG. 7 provides an overview of data flow between logical servers;

FIG. 8 shows catalog enablement;

FIG. 9 shows content librarian workflow;

FIG. 10 shows image librarian workflow;

FIG. 11 shows a protocol support for filemove; and
FIG. 12 shows a distributed procurement service.

DETAILED DESCRIPTION OF THE INVENTION

The disclosed invention automates much of the manual processing described above. The disclosed invention preferably is used in the following way (FIGS. 3, 4 numerals 14, 16 illustrate preferred steps involved):

1. An employee 17 preferably accesses one or more electronic catalogs 24 stored on a shadow catalog server 22, accessed via a local area network 20 preferably by means of a employee workstation 18. These catalogs contain only those items for which a price has been negotiated between the enterprise and a particular supplier, so the verification by the enterprise's Purchasing department described above is obviated.

2. The employee selects items from the catalogs preferably with a mouse or similar device. Catalog items may be displayed with pictures, descriptions and other information in a fashion similar to a paper catalog. Where similar items are available, a "Compare" icon can be selected on the screen, causing the items to be listed side by side, with differences highlighted. Items can be located by searching down the taxonomy tree of the catalog (much as one searches through a paper catalog by finding the appropriate general section and then looking for a particular item), or by entering a search word or phrase.

3. Items selected may be accumulated in a "clip-board", a temporary holding area on the user's computer disk. When all required items have been selected, the employee selects a "Submit" icon. This causes the selected items in the clip-board to be sent to the appropriate approvers as a Purchase Order 30. It should be noted that there is no manual transcription of ordering information from the catalog to the purchase order (since that is performed by the disclosed system).

4. After the order has passed through the enterprise's normal (legacy) business systems, including a workflow definition database 26, a purchase order database 28, and other existing corporate applications 32, it is forwarded to the Maintenance Entity via the Network 34. From there it is sent to the supplier for fulfillment in a traditional way.

5. The employee can at any time review the purchase orders already submitted; cancel them if they have not yet been shipped; and print reports.

The other end of this process involves capturing catalog content (text and images as well as price information) in electronic form. As long as catalog content providers and suppliers continue to publish traditional paper catalogs side by side with the electronic versions, this activity does not displace the current manual process.

The two ends of this automated solution, the supplier side and the buyer side, may be brought together by means of a computer network and associated service offerings. FIG. 6, shows an overall network topology 20 or the initial implementation, with the three major pieces (supplier, buyer and Network Central) clearly distinguished. Suppliers 50 provide e-mail directly to the maintenance entity 100 and to the communications gateway 82 via an EDI mail box 110. Suppliers also provide CD's and diskettes for the terminals 52 and 54 for the purpose of catalog creation and maintenance. Terminals 53 and 54 communicate with the maintenance entity through Token Ring THORNWOOD 56.

Maintenance Entity 100 consists of a local area network backbone 98 which supports multiple token rings 76 and 94

in addition to a communications gateway computer 82. Token ring 76 services catalog file and reference services computer system 72, catalog and software distribution server 74, and PS/2-OS/2 78. Token ring 94 services system and file monitor server 96 and PS/2-OS/2 92. The local area network backbone 98 further supports and EDI gateway pricing decision computer 80. Catalog maintenance activities are input to the Maintenance Entity 100 from remote terminals 52 and 54 via token ring 56 through T-1 leased lines 62. Application development occurs remotely, item 58, and is communicated to the operations environment through application development Token Ring TORONTO 60 and routed to development server 84.

The client environment is shown in the lower segment of FIG. 6, defined by shadow server 106 which maintains a customized copy of the master catalog for distribution to local clients 102 and 104. Purchase orders are received by a Local buyer master server 86 from a data pathway connecting remote shadow LAN 108 with local buyer master LAN 88. The Buyer Master Server also performs the server function in the following capacities; order processing from buyer clients 90, approval and call back. The Buyer Master Server communicates with the operations environment of the enterprise through a 56 Kb switched or leased TCP/IP line 107.

The Buyer Master Server also interacts with the Range of Legacy Systems which consists of a host computer 64 connected to a COMM FEP 66, a MINI computer 68, and a PC 70. The services provided by the Maintenance Entity distinguish this invention from the electronic catalogs offered by public networks such as Prodigy (tm) and Compuserve(tm), which do no more than route messages from the subscriber to the supplier. The Maintenance Entity services act as a single point of contact to all the suppliers on one side and all the buyers on the other. Existing networks merely link them in a many-to-many relationship. The advantages of this approach are clear: for example, a supplier has only to provide a change to a catalog item once to the Maintenance Entity, which then disseminates it to the affected users.

This aspect of the invention is summarized in FIG. 7, numeral 22 which shows the data flows already mentioned in FIGS. 3, 4 as well as some others which were omitted from the original description for the sake of clarity. The omitted data flows include details specific to both the Operations environment and the Client environment:

1. Details of the Operations Environment 125 Networking software and services software including; a PRICING DAEMON 132 which receives purchase orders, change orders and cancellation order acknowledgments from the ORDER PROCESSING SVR 154 located in the Client Environment 123 via an EDI GATE Way 130. The Pricing Daemon 132 in turn provides pricing updates and base catalog entries to catalog file servers, CAT FILE SERVERS 140.

The operations environment further includes distribution management tools, defined generally by CMVC 134. The CMVC consists of a Subscription List 138 which resides on a server in the form of topology tables 136. The subscription list sources multiple reference servers 142, all of which are contained within a distribution server 144. The reference servers source a combination server comprised of a system monitor server 146 and a file mover server 148.

1. Details of the Client Environment 123

Comprised of a Shadow Server 150 consisting of Browser Dynamic link libraries DLLs 152. The Browser DLLs receive catalog data from the Order Processing Server 154

and in turn output the Browser DLLs and customized catalogs, during a client browse session to a buyer (client) 156.

The Order Processing Server receives inputs from four separate sources; (1) Buyers (clients) 156 (2) the Approval Server 158 (3) the CallBack Server 160 which services the transfer of files to and from legacy systems 164 and (4) the File Mover Server 148, which is part of the Operations Environment.

This aspect of the invention preferably comprises (see FIG. 7) three major components:

1. Catalog creation and maintenance tools (shown at the top of Fig. 7). Catalog creation is defined by item 122, the SELLER AND PROVIDER ENVIRONMENT consisting of EDI MAIL BOX 122, CONTENT PROVIDER 124, and CD's & Diskettes 126.

Catalog maintenance is defined by item 127, CATALOG MAINTENANCE ENVIRONMENT, which includes item 128, CATALOG MAINTENANCE CLIENTS which receives inputs from CDS & Diskettes 126 and additions and changes concerning catalog entries & update, pricing updates, and subscriptions from CAT FILE SERVERS 140.

2. Catalog browsing and purchasing software (the client environment shown in the lower segment of FIG. 7); and,

3. Networking software and services (the Operations environment shown in the middle segment of FIG. 7) defined by OPERATIONS ENVIRONMENT 125.

Catalog Creation Maintenance

The preferred embodiment is made up of two main elements:

Content management tools to receive, process, and manage images 208 and text 212 from content providers 200 for the creation of an EPS (Electronic Purchasing Service) master catalog. An overview of this process is shown in FIG. 8, numeral and Text 212 from content provides 200 are first converted through conversion units 210, 214 also, including conversion units, 218 and 222 from third party converters 202, the graphics and text are then and combined with content from independent image providers 220 to create catalogs 216 and 224 constituting third party catalogs 204 which are then combined at an EPS catalog stage 206 to form EPS (Electronic Purchasing Service) catalog 226 and distributed to buyers 230 via EPS subscription 228;

These enable EPS Operations staff to create and manage catalog information in the merchandise database such as the price, description, and visual representation of each item.

Distribution management tools to receive vendors' price and catalog updates, as well as to propagate the changes to the customers' Buyer Master servers.

Content Management Tools

The EPS Content Management tools preferably comprise:

FotoFarm;

Product Editor;

Folder Editor;

Subscription List Editor;

NAM2DAT/NAM2GRP.

FotoFarm

This collection of utilities may be used to convert text and images from the content providers 200, 250 and 280. The workflows of these two activities are shown schematically in FIGS. 9, 10 numerals 26, 28. Supported functions may include:

Receive, store, and archive source images 282 and text files 252 and 282.

First-level validity check of source media 254, 284 and 286.

Assign EPS unique filename and update the index files 258, 284.

Create master catalog's subchapters and folders, and populate them with the relevant contents 260 292.

Trigger down-stream re-creation or subscription catalogs (see below) when EPS catalog updates occur 260 292.

Process images received from content providers in batch model 256:

Delta cropping of image by specifying new crop coordinates 288.

Generate multiple resolution versions of images.

Convert 24-bit to 8-bit dither with palette matching.

Enable re-scheduling of batch process for related information.

Allow multiple images to be proofed at the same time.

Manage registries of 260 292:

Shared disk storage for various purposes;

Providers of images, text, or EDI content and services;

All top-level books in the EPS Catalog Server.

Product Editor

This ITS (Iterative Transaction Systems) application enables EPS Operations staff to:

View multiple product descriptions at a time;

Associate images with product handle;

Save, import, and create templates;

View and edit product descriptions.

Folder Editor

A sample screen from this tool is shown as FIG. 11, numeral 30. This tool enables Operations Staff to:

View master catalog space.

Organize and arrange products into groupings, i.e., manipulate Catalog topology.

Search: Keyword, Power Search (Attribute, Taxonomy).

Subscription List Editor

This uses code from the Folder Editor and Search Engine, with additional functions, to enable Operations staff to:

Save subscription profiles;

Update/edit Distribution Frequency via manual operation;

Update/edit Distribution Frequency automatically;

Generate flat or hierarchical browser space;

Send newly created subscription list to Catalog Server.

NAM2DTA/NAM2GRP

NAM2DTA is a stand-alone GML parser that converts tagged source files to catalog objects. NAM2GRP is a stand-alone GML parser that converts tagged source files to catalog folders.

Distribution Management Tools

Distribution Management concerns itself with getting data from Network Central out to the customers' computers on the network in such a way that each computer receives only that data (catalog items and price information) which it needs. The EPS Distribution Management tools preferably comprise:

Distribution Manager with GUI query tool;

Catalog daemon (CATD);

FileMover.

Distribution Manager

Responsible for

Price Update (Catalog Monitor);

Catalog Update (Catalog Monitor);

Automated EDI Processing and Distribution (Update Daemon);

Acknowledgement Processing;

Update DB2/2 Tables.

Catalog Daemon (CATD)

This software runs in customers' servers and polls mailboxes to apply updates, and preferably monitors channels for action objects including: Images; Applications; Prices; Catalog descriptions. It preferably can Execute action specified in action object; Forward acknowledgement objects to parent; and is Used together with FileMover daemon to verify file movement.

Filemover 300

This is the communication layer of EPS which moves files between systems. In particular, it drives the movement of data throughout the network for price up-dates and purchase orders.

Filemover uses a simple mail-like mechanism in which files are moved from OUTBOX directories to INBOX directories, which can span network attached systems. It has no knowledge of the type or structure of the file it moves, and treats them all as binary blob data. Hence no code page translation or character set translation is attempted.

Filemover is a Point-to-Point protocol and does not provide any internal routing mechanism. Routing can be provided outside of the Filemover, however, by:

Utilizing the routing mechanisms of the underlying protocols, e.g., TCP/IP. Will only work across machines running in a single internetwork with the same protocol.

Using Catalog Daemon (CATD) to provide routing by forwarding files across intermediate nodes. It works across internetworks with different communication protocols but its network path must be hard-coded within the catalog files being sent.

Filemover enables the EPS to:

Move files of any size and takes care of splitting and reassembling the files when the underlying communication software has a limitation on the file size.

Verify and confirm both file movement and ability to move files (e.g., checks disk storage).

Support file movement over systems connected with multiple protocols (shown schematically in Fig. 11):

IPX/SPX 306

NetBIOS 304

TCP/IP 310

Support file movement over:

Dial-up connection (SLIP and LAN Distance 312) via modems

Dedicated LAN or WAN connection

Support Store and Forwarding of files 302.

SNA APPC 308

Client Environment

Recall that the Client Environment (FIG. 7) comprises two principal components:

1. An electronic catalog in a format that can be browsed, searched and ordered from, by a corporate employee with no training in Purchasing procedures;

2. Software that controls the flow of a purchase order through an enterprise's procurement procedures.

The preferred embodiment of the above software and related manual processing minimizes data storage require-

ments and maximizes the responsiveness of the total system preferably by employing a primary model consisting of two major components, a buyer side component 330 which communicates over a network bridge with a premises component 332 (see FIG. 12, numeral 32):

1. An end-user computer system 333 attached to a Local Area Network or LAN, containing the usci interface;

2. A "shadow catalog server", 331 (FIG. 12-2) with disk storage that can be accessed over a LAN by one or more end-users' computers, the disk storage being used to hold one or more electronic catalogs 328 and program code 331 to enable browsing of the catalog and transmitting purchase orders to the "buyer master server" 324;

3. A "master buyer server" 324 (FIG. 12-3), which is a computer system within the enterprise containing (1) program code (described below) which can take purchase orders 332 from one or more end-user computers and control their flow through the enterprise's business processes, as described under "Workflow" below, before transmitting them over a network to the supplier via the Maintenance Entity 320 and (2) a to a Purchase Order data base 322 that can be accessed over a LAN 326.

A preferred embodiment of the above comprises:

Order Manager and Catalog Browser

This function runs on the end-user's personal computer, although the code would normally reside on disk storage in a catalog shadow server machine. It provides the following main function to an employee using the system:

Log on/Password Security

30 Login

Log into EPS

Track User ID for all transactions arising from this session.

Additional Order Manager functions may be enabled or disabled based on the login profile.

User selects from list of authorized users.

Catalog Browser

Browse Product Images, Text and Prices

40 Able to page forward or backward.

Quick return to top menu page from any part of the catalog.

Quick return to the table of contents from any part of the catalog.

45 Display previous page at top of screen, with links to navigation log.

Images are displayed in .BMP format.

Two separate image files are kept for OS/2 and Windows.

50 See also "FotoFarm", supra.

Text The Browser may select zero, one, or more ordered sets of descriptive phrases.

Prices.

Select Product Based on Single Keyword

55 Based on index search.

Index search is launched with user's action on an icon represented by a magnifying glass.

Search by product type or manufacturer's name.

60 Copy to clipboard for further processing.

Compare Products (max. 4)

Compare up to four items from the same category.

End user selects this option by clicking on a "Compare" icon represented by a scale.

65 Varying features amongst the compared products are highlighted in bold text.

Product Clip Board

Select items on Product Listing for adding to clipboard.
 Add item on Product Page to clipboard.
 Delete an item in the clipboard.
 Change the quantity of an item in the clipboard.
 Clear the clipboard to remove ALL items.
 Save the clipboard (to a file).
 Submit the clipboard (as a purchase request).
 Show the items on the clipboard.
 View clipboards (i.e. saved clipboard files).

Purchase Request Generation Select the recipient of the purchased items from a list. The recipient list is kept in a local disk, and its entries can be added, changed, or removed.

Classify all line items into capital and expense items.
 Ask for budget number for capital items.
 Ask for engagement number for expense items.
 Display generated purchase request for confirmation.
 Select approver from list to submit request to.
 Add comments to purchase request.
 Send purchase request to approver.
 Save purchase request as clipboard.
 Cancel submission of purchase request.
 Print Clipboard: This function is in addition to, and separate from, the report generation functions which use DB2/2 report generators. It enables users without access to DB2/2 to print a clipboard or submitted order from their own workstations.

The supported functions include:

Generate printed output from client application for the contents of the clipboard.

Can also be used for printing a submitted order.

Able to generate output for various printer formats including Postscript. May have to go through some type of metafile generation process.

Needs to work in both OS/2 and Windows.

Purchase Order Creation**Multiple Sellers on One PO**

Each line item in a purchase request could be sent to a different vendor. This requires that information such as the shipping and billing address be stored on a line item level, rather than at the header level for a purchase order. These multiple sellers include those whose products are not listed on the catalogs.

Electronic PO

This is to forward the purchase orders electronically to the vendors via the EPS. system. Data includes type of transaction, required data as defined by EDI standards for a 850 PO such as PO number, date, name & address, customer ID, customer master record for shipping and billing information.

Print Paper PO

Print PO by vendor in a multi-vendor PO

Sub-function of the common printing functions described in "Report Generation", infra.

PO Processing**Manual Status Update**

Purchaser can update status of PO manually after receiving acknowledgements, status updates, etc. from vendors via fax, phone, or mail. Changes to the PO can then be saved to the DB2/2 database on the Purchasing Server.

Line Item Modifications by Client

The quantity of line items can be increased or decreased prior to order submission. After an order has been submitted, the quantity can only be decreased.

The quantity of line items can be decreased to zero.

A line item can be deleted once an order has been placed with a vendor and is being fulfilled by the vendor.

Allow client to modify other fields such as requested ship date, shipping and billing address, add comments to line items (e.g., a banking institution).

Possibly allow client to switch partnumbers, delete line items, add new line items.

Change Logging/Reporting

Changes to the POs are recorded in the logs and can be accessed by the report generation functions.

PO Maintenance**Browse POs**

Group existing POs in chapters with summary information including:

Request number.

Requester.

Recipient.

Request date.

Total price.

Line of Business.

Scroll through all line items.

Sort line items by column headings in the following order:

Sort numeric columns from high to low or low to high.

Sort alphabetic columns from A to Z or Z to A.

View details of line items by clicking on them.

Search for specific groups of POs and purchase requests by Requester Name, Requester Date, and Request Number.

The search results can be grouped into a chapter.

View Approvers

Enables user to look up list table of associated approvers for a PO.

Approvers are item-based; i.e., each item has its own approver.

Enables user to view approval data for each item, with the approvers name, decision, and date of action.

Check Status

Enables users to check current status of POs. * When orders are placed, vendors send acknowledgements and status messages via EDI. These are reflected in the updates to the status of line items, with the date of the status change.

The approval status of each order or request can also be checked.

Cancel POs

Enables users or administrators to cancel POs or purchase requests. This is dependent on whether the order has passed the deadline for the change to be effective. Vendors restrict the set of order states against which a PO can be canceled. For example, if the item has been shipped, the order cannot be canceled. EDI Vendors must be able to support Cancel/Change 860 transactions and their subsequent acknowledgements.

The end user will be prompted for confirmation of cancellation request prior to processing.

Upon confirmation of cancellation, the order is moved to the Canceled Requests chapter.

Note

If the PO has already been sent to the vendor, an additional EDI (860) transaction is generated and budgets credited.

Add Approvers

This function is required for customers who are not using e-mail approval.

Requires end-user interface to be modified to make the approver line items editable.

Changes must be saved back to the database.

Delete Approvers

This function is required for customers who are not using e-mail approval.

Order Manager Administration**Budgets**

All line items for capital purchases are charged against the capital expense budgets.

Budget Information: Each budget has the following information:

Budget number

LOB (line of business)

Line of business related to the purchase.

Allocated

Amount allocated to the budget.

Current Requisition

Amount associated with all active orders.

Ordered

Amount for orders that have been sent to vendors.

Balance

Balance amount available in the budget.

Active

Yes for an active budget.

No indicates a budget that has been closed and no expenditure can be charged against it.

Create new capital budget: The following information is preferably required:

Budget number

Amount allocated

Amount allocated to the budget.

Active status

A Yes to indicate that it is active.

LOB (line of business)

Line of business the budget is assigned to.

Description of budget

Brief description of the purpose of the budget.

Change existing budget: Changes can be made to the following aspects of the capital expense budget:

Amount allocated

Amount allocated to the budget.

Active status

A Yes to indicate that it is active.

Description of Budget

Brief description of the Purpose of the Budget.

All changes are logged to the change history log, which can be viewed.

Delete existing budget: This function may be used to clear existing budgets at the end of a fiscal year.

There will be a prompt for the user to confirm deletion of the budget amount.

View budget history

All changes to the allocated amount and status of the budget, as well as all expenditures are logged.

Each transaction is logged with a brief description of the activity attached.

The last 100 transactions are logged.

This view function is applicable to both active and inactive budgets.

Import budget guidelines: This is to enable the import of budgets from SAP, or another accounting interface, for the initial budget creation.

Track budget: All line items in a PO get charged to either an expense or capital budget. Any modifications to the PO requires that the budget balances get updated.

User Profiles

The following functions are provided:

Add new users Authorize new users to the list of authorized users.

Delete User

Update current user profile

Ship/Bill Update (Per Site)

This is to enable the purchasing administrator to maintain the shipping and billing addresses of all locations within a Customer enterprise. These addresses are referenced by POs to define the shipping and billing destinations. The supported functions include:

Add new address;

Update current address;

Delete current address;

Vendor Info Update

This is to enable the purchasing administrator to maintain the EDI vendor addresses for confirmation of shipment and billing, as well as status updates. The supported functions include:

Add new address;

Update current address;

Delete vendor address.

Define Company Policy For Capital/Expenses Purchases Line of Business

Add new line of business;

Update line of business information;

Delete line of business.

Report Generation

This includes the common printing functions for both the reports and purchase orders. They include:

Commodity reports;

Line cost reports;

Order modification reports;

Budget modification reports;

Customized reports.

Approval Workflow

Approval workflow is controlled by the Approval Manager residing in the Purchase Server in the customer's site. This workflow of the purchase orders between the customer and vendors is enforced by a PO approval process defined by the customer. Its functions include:

Keep track of a PO's approval status from the moment a purchase requisition is generated. Appropriate actions are taken to forward the purchase requisition to the predefined approvers to be approved or rejected.

Interface with the customers' electronic mail systems to post approval notifications for the necessary action by designated PO approvers.

Provide separate ITS client application to allow PO approvers to approve purchase requisitions directly from within the EPS system rather than from the external email system.

Approval Policy Configuration

Set up Lotus Notes DB to specify approver hierarchy.

Use of REXX code to customise approval hierarchy.

Approval Data

Store approval data for POs in DB2/2 PODB;

Store list of approvers in Lotus Notes;

Entry point API (call-out) to support accessing approvers from external systems.

Approval List Generation

Print approver Lists from Lotus Notes;

View approver Lists from Lotus Notes;

Lotus Notes interface to create and update approvers list (company wide);

REXX code to specify approval policy given PO and list of approvers;

Generate Approvers service application 158 to communicate with client server API CORDER PROCESSING SVR 154 (per PO).

Approval List Processing

Approvers can receive email messages via an E-MAIL SERVER 162 notifying them that they need to approve various purchase requisitions.

Mail routing support is available for Lotus Notes, cc:Mail, and Microsoft Mail.

Snapshot Database Within Lotus Notes

This is a function to synchronise the information between the EPS Server and the Lotus Notes server.

Help Functions Within Lotus Notes

A help database is available from within Lotus Notes as a Notes database.

Approval Manager Client

This ITS client application is for approvers who do not want to receive approval notification from an email system external to EPS. It enables approvers to log on to the EPS Purchasing Server and approve or reject purchase requisitions.

PO Workflow

The flow of the purchase order through an enterprise's approval and other financial processes varies with each enterprise. The disclosed system contains workflow logic implemented as a Finite State Machine. This is a table specifying how the system is to change state in response to specified inputs, and what actions it should take when each transition takes place. Such a table can be easily tailored to fit the needs of a particular enterprise. Application Program Interfaces (APIs) in the generic state transitions supplied with the system allow an enterprise to invoke and pass information to and from existing computer applications and data bases (which could include the enterprise's "legacy" purchasing system) as shown in FIG. 4 step 02.

In the preferred embodiment, The EPS Client/Server application programming interface (API) provides client applications with a set of functions and action calls to communicate with the EPS Server for managing purchase orders within the customer's environment. It can also be used by any customer applications to work with the data available from the server.

The API supports three types of client applications:

Interactive Transaction System (ITS) applications

These are clients written with the ITS toolkit and using the ITS runtime to provide the user interface.

Non-ITS applications

These are clients that have user interfaces other than ITS.

EPS system extensions

These are ITS and non-ITS applications registered with the Purchase Order workflow of the EPS server, and can be classified into two categories:

1. EPS Monitors

Registered against a certain state in the Purchase Order workflow and are notified whenever any purchase request enters that state. The notification will be received asynchronously with the purchase request continuing within the workflow.

2. EPS Services

Registered with the EPS Server and introduce a new state in the Purchase Order workflow. They are notified whenever any purchase request enters that state, and are

expected to notify the EPS Server when the specific task is completed.

API Architecture

A brief description of the API architecture is necessary for an understanding of the API functions and action calls. Essentially, the API is made up of five layers, as shown in FIG. 5:

X layer 38 This is the ITS application layer which interfaces with ITS client applications to deliver ITS client application requests to the C layer for onward transmission to the Buyer Master.

Y layer 40 This is the non-ITS application layer which interfaces with non-ITS applications to deliver non-ITS client application requests to the C layer for onward transmission to the Buyer Master.

C layer 42 This is the common layer that interface the X and Y layers with the M layer. It takes application level data structures from X or Y layers, converts them into low level communication data buffers, and invokes M layer functions with these data buffers to communicate with the EPS Server.

M layer 44 This message layer is the lowest level EPS API communication layer. It uses the ITSCOMM 48 API to handle protocol specific communication functions. It handle communications data buffers instead of application level data structures.

S layer This is a server interface layer and handles functions between the C layer of EPS Server and the OM Server layer 47.

In addition, the EPS Client/Server API has a set of OMServer APIs to support client requested actions on the EPS Server from the S layer 46.

APIs

Logically, the EPS Client/Server APIs can be grouped into four areas:

Information

These include the functions and actions to Get and Save data of predefined datatypes to the EPS Server.

Monitors

Services

Security

These validate clients and restrict the level of access. These include: Logon, Logoff, Connect, and Disconnect.

THE NETWORK ENVIRONMENT

"Network Central" (shown in the middle of both FIGS. 6, 7) consists of one or more computers and program code of the type located in IBM's Advantis network, which provide four principal functions:

1. receipt of purchase orders from the master buyer servers of one or more enterprises;
2. passing EDI transactions to and from suppliers;
3. storing catalog information for transmission to buyers;
4. the distribution of price information.

In addition, personnel at Network Central may perform many or all of the activities described under "Catalog Creation/Maintenance" above.

Overview of EDI

The Electronic Data Interchange (EDI) is a standard for the exchange of business data. It defines:

Communication wrappers, which are usually handled by a communication package from a Value Added Network (VAN) providing EDI mailboxes).

The ASCII character set.

Various transactions, each with an ID

The order and hierarchy of data within each of the transactions.

The type and length restrictions for each piece of data.

There are two major EDI standards—ASC X12 which is the standard for the United States, and UN/EDIFACT, which is the general standard adopted by countries outside of the U.S.

EPS EDI Implementation

Customers may be connected to a system like the IBM &EPS. via non-EDI links, sending their purchase orders over 56kbs lines. The EDI Gateway translates and maps these communication into EDT messages before using the FileMover mechanism to send them to vendors via its EDI mailbox on the Advantis network. Vendors' acknowledgements and status updates are transmitted as EDI messages to the Advantis mailbox. The EDI Gateway translates them to the EPS format and updates the PO status accordingly.

Currently, the EDI Gateway consists of the EDILISTS application as well as the IEBASE program provided by Advantis to connect to the Advantis network using LU6.2 communication protocol. The gateway logs and tracks all EDI transactions for diagnostic purposes; it can also alert the Network operator for the necessary recovery action when it encounters any unexpected event.

The preferred embodiment supports the following EDI transactions:

832-Create Catalog Content.

850-Purchase Order.

855-Order Acceptance.

856-Shipping Order Status.

860-Change/Cancel Order.

865 and 870-Order Status.

997-Acknowledge Receipt of Order.

Other Gateway Functions

Gateway EDI in-box and PO in-box monitoring

Look for new PO request and translate to EDI.

Look for new EDI transaction and translate to EPS internal format.

Poll EDI mailbox hourly if not done via PO transactions.

Execute utility to access Advantis's mailbox using

IEBASE

Upload EPS-generated EDI to VAN.

Download any waiting EDI message.

Directory browsing.

PO lookup, browsing and merge update. Look up and browse archived PO files.

PO/EDI error logging.

Reports all errors and alerts system operator for attention in the event of a major error taking place.

Logs problems to a flat file.

Event tracking and logging.

Catalog tools linking for 832 transactions.

Archiving POs to multiple customer/vendor file systems. What is claimed:

1. A system for electronically ordering items within an enterprise comprising:

a maintenance entity, located outside said enterprise, for receiving price and availability data from a plurality of distributors and means for receiving and processing images and text of catalog data from a plurality of catalog content providers for creating and maintaining one or more electronic master catalogs in a central location for subsequent distribution to a plurality of shadow catalog servers distributed throughout said enterprise, where said shadow servers contain data

representing a customized electronic catalogue from said master catalogs, over a computer network, said maintenance entity comprises:

means for constructing said customized catalog from said master catalogs;

means for receiving a supplier's price and catalog availability changes from a plurality of said distributors and propagating them to one or more selected buyers over said computer network;

means for receiving catalog changes from a plurality of said catalog content providers and propagating them to one or more selected buyers over said computer network;

one or more computer systems, maintained by said maintenance entity, and having means for creating and transmitting to a plurality of shadow catalog servers within said enterprise, images and text of a plurality of catalog items offered by said content provider and price and availability data from said suppliers; said enterprise comprises:

a plurality of first end-user computer systems geographically distributed throughout said enterprise comprising a user interface and able to access disk storage on said shadow catalog server, and having means for electronically ordering said catalog items from said shadow catalog servers;

said shadow catalog servers, which comprise a second computer system whose disk storage can be accessed over a local area network by one or more first end-user's computers; said disk storage being used to hold and maintain (1) one or more customized electronic catalogs, and (2) internal program code comprising a Catalog Browser capable of transmitting purchase orders to a master buyer and server, and having means for allowing an end user to view said customized electronic catalog in order to facilitate comparisons between catalog items from several suppliers, and in permitting orders to be generated containing items from several suppliers;

said master buyer server comprising a third computer system located within said enterprise containing (1) purchase order workflow software comprising an order manager and a purchase order workflow which takes purchase orders from one or more end user computers and controls their flow through said enterprise's business processes before transmitting them over a network to the supplier; and (2) a Purchase Order data base; and

means for said master buyer server to receive information from and transmit information to said supplier.

2. A system according to claim 1, further comprising a means for specifying, subscription information comprising a mapping between a plurality of supplier's catalog items and one or more buyers for indicating which catalog items can be propagated to each buyer's catalog server, where said mapping is unique to each said buyer's catalog server.

3. A system according to claim 1, further comprising a means for specifying, pricing information comprising a mapping between a plurality of supplier's catalog items and one or more buyers for indicating which catalog items can be propagated to each buyer's catalog server, where said mapping is unique to each said buyer's catalog server.

4. A system according to claim 1, further comprising an EDI gateway wherein EDI transactions between one or more distributors and said maintenance entity are converted to

17

pricing, catalog availability and purchase order transactions between one or more customers and said maintenance entity, and wherein EDI transactions between one or more suppliers and said maintenance entity are converted to catalog update transactions between one or more customers and said maintenance entity.

5. A system according to claim 1, further comprising a means for enabling said purchase order workflow software to be mapped to the business processes of an enterprise.

18

6. A system according to claim 1, further comprising mapping tables constructed by said maintenance entity for identifying suppliers, manufacturers, content providers, catalogs, catalog items and buyers for uniquely identifying each, whilst allowing said suppliers, manufacturers, content providers and buyers to continue to use the naming conventions established within their organizations.

* * * * *



US005825651A

United States Patent [19]

Gupta et al.

[11] **Patent Number:** 5,825,651[45] **Date of Patent:** Oct. 20, 1998[54] **METHOD AND APPARATUS FOR
MAINTAINING AND CONFIGURING
SYSTEMS**[75] Inventors: Neeraj Gupta; Venky Veeraraghavan;
Ajay Agarwal, all of Austin, Tex.[73] Assignee: Trilogy Development Group, Inc.,
Austin, Tex.

[21] Appl. No.: 707,187

[22] Filed: Sep. 3, 1996

[51] Int. Cl.⁶ G06F 1/00

[52] U.S. Cl. 364/468.09; 364/488

[58] Field of Search 364/468.09, 468.02,
364/468.03, 468.1, 488; 395/651[56] **References Cited****U.S. PATENT DOCUMENTS**

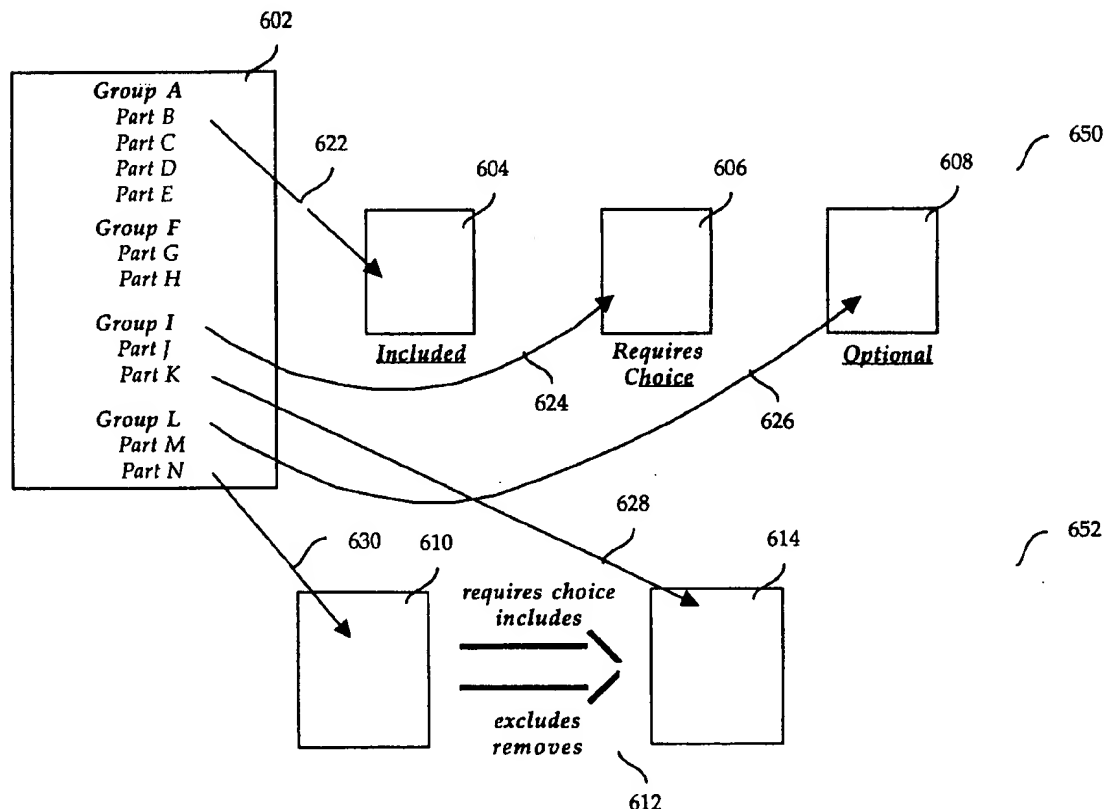
4,796,194 1/1989 Atherton .
5,019,961 5/1991 Addesso et al. 364/192
5,019,992 5/1991 Brown et al. 364/468

5,355,317 10/1994 Talbott et al. 364/468
5,357,440 10/1994 Talbott et al. 364/467
5,586,052 12/1996 Iannuzzi et al. 364/512
5,659,478 8/1997 Pennisi et al. 364/468.01

Primary Examiner—Kevin A. Kriess
Attorney, Agent, or Firm—Hecker & Harriman

[57] **ABSTRACT**

The invention provides the ability to interactively select and configure a product among a set of related products based on availability and compatibility of features and options. It does not impose an order in the selection of products, features or options; only valid selections can be made at any time. To create an electronic representation of the product information to achieve the above goal, the invention provides a framework for defining a systems by defining the components of the system using elements contained in a parts catalog and defining relationships between the components of a system. A configuration system validates a configuration using the system definition, the current state of the configuration and user input.

85 Claims, 24 Drawing Sheets

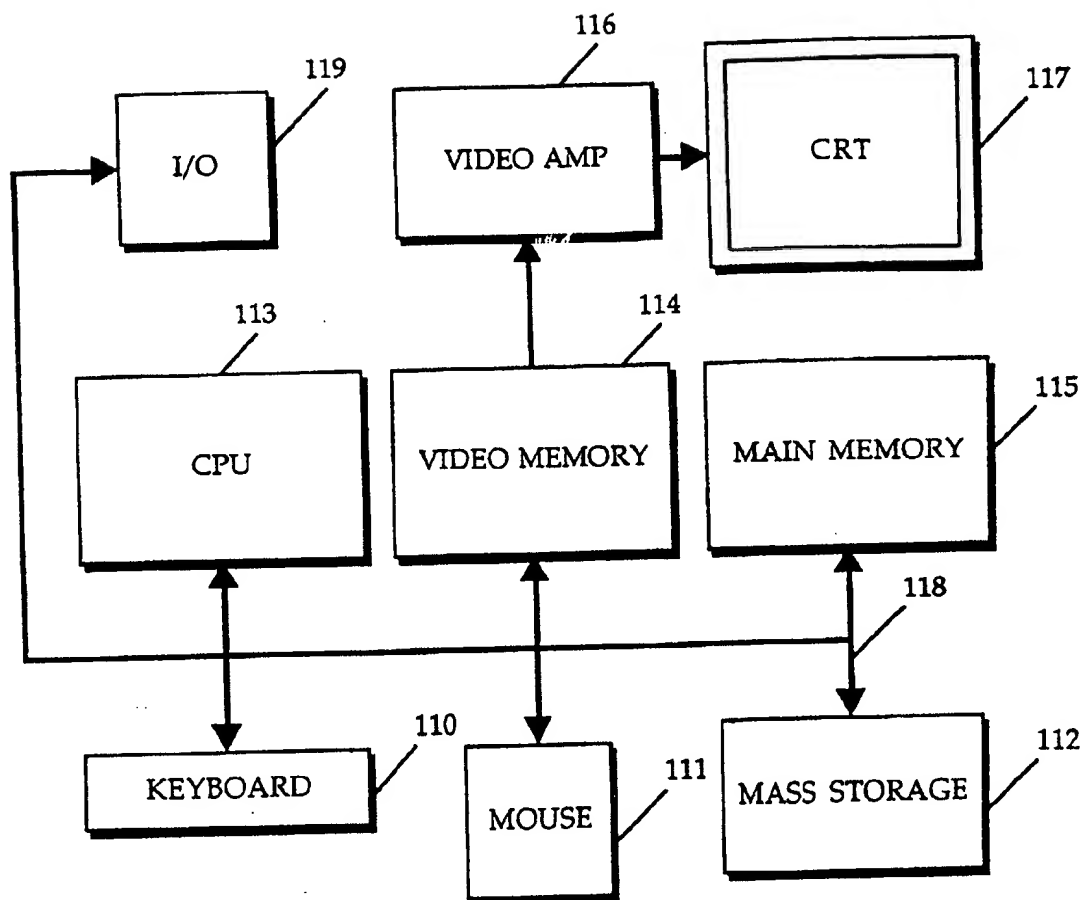


FIG. 1

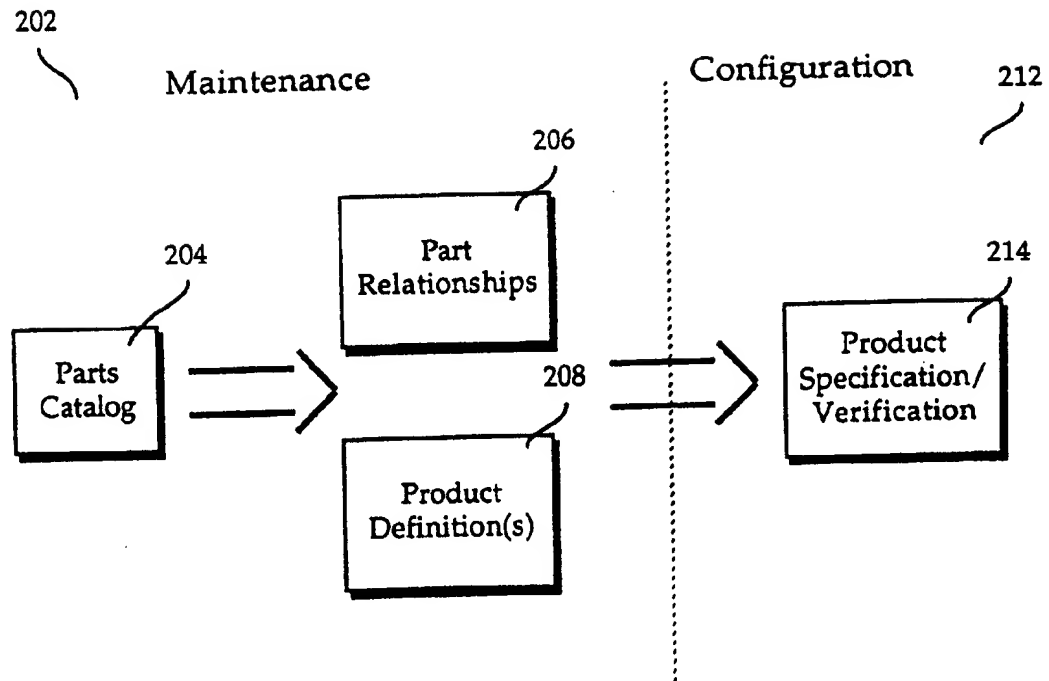


FIG. 2

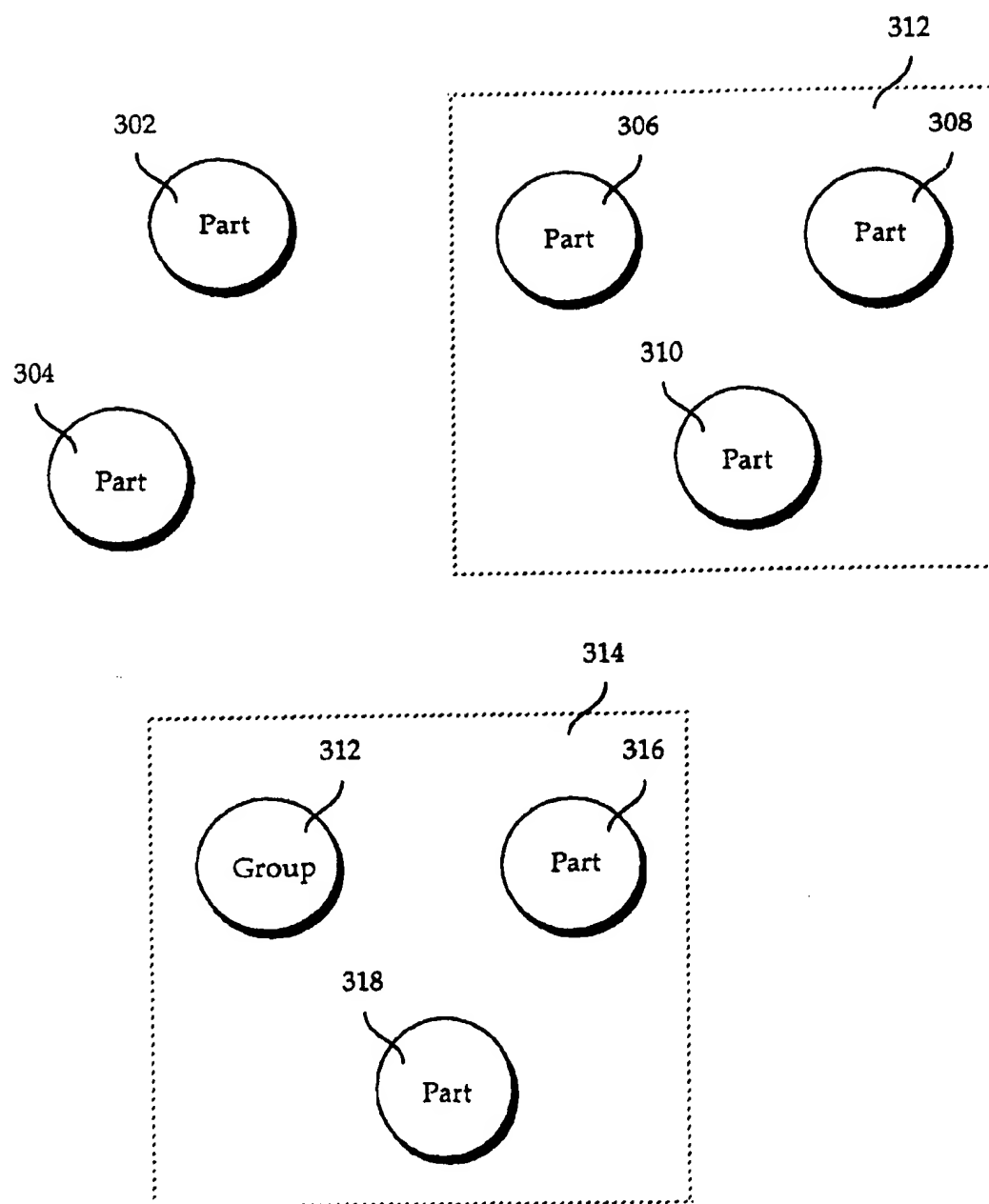


FIG. 3

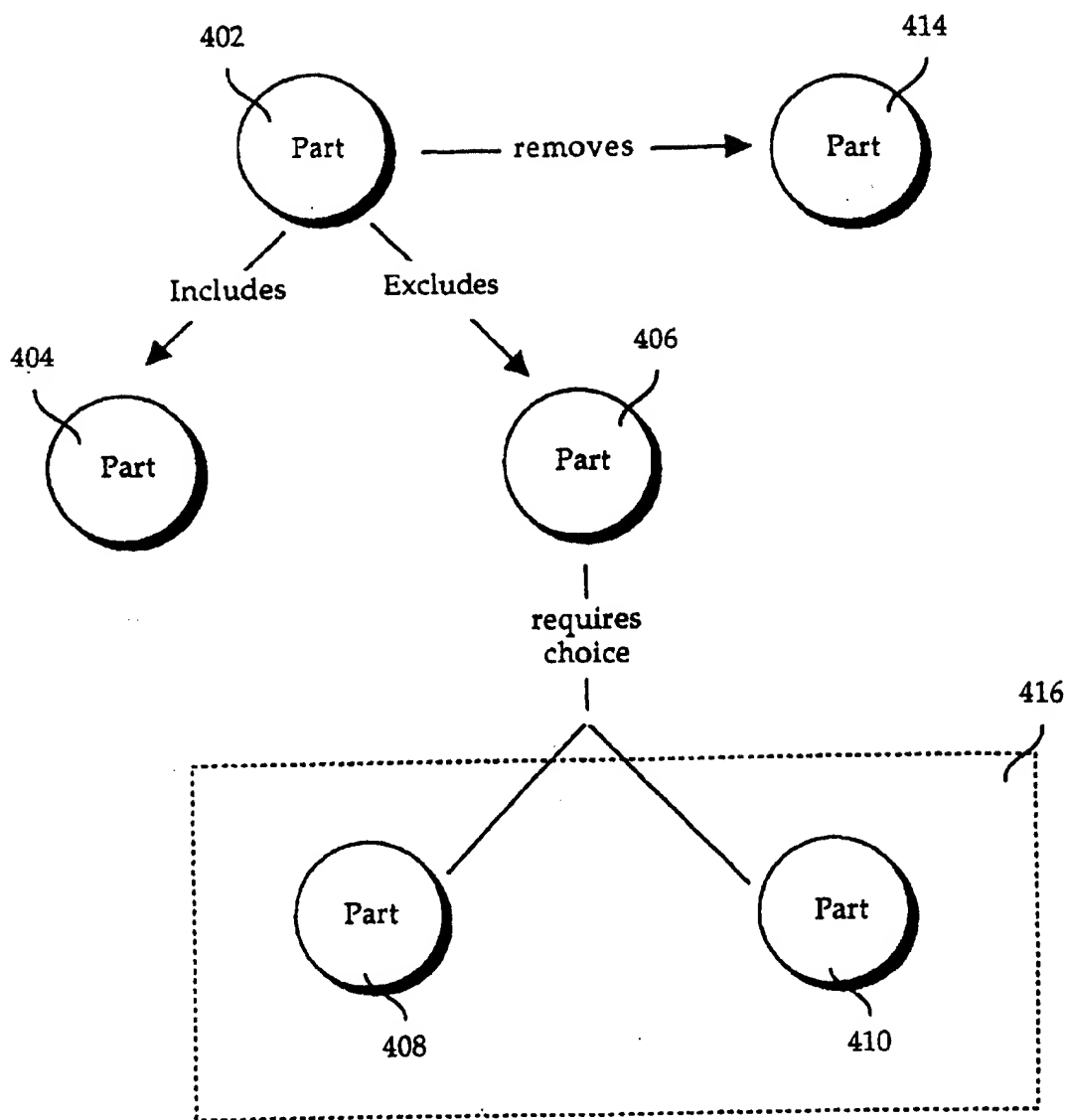


FIG. 4

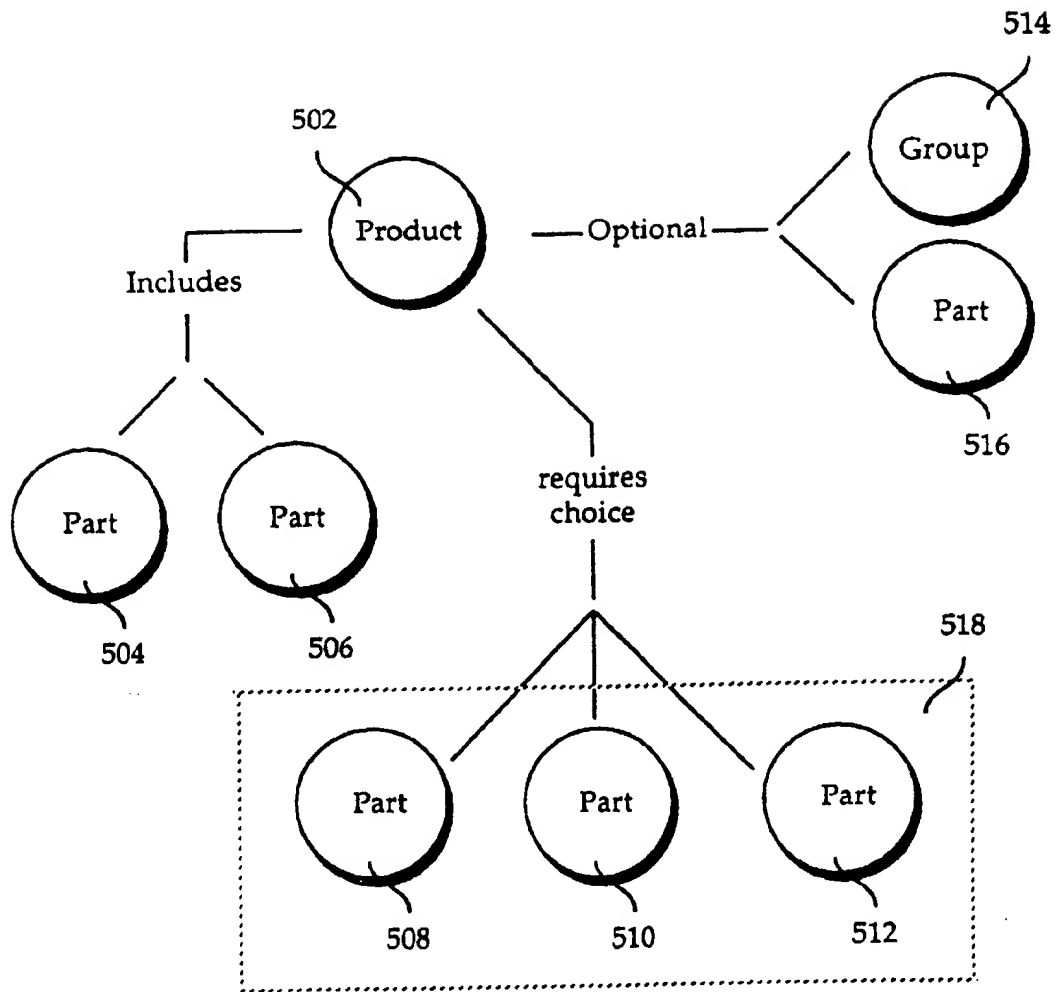


FIG. 5

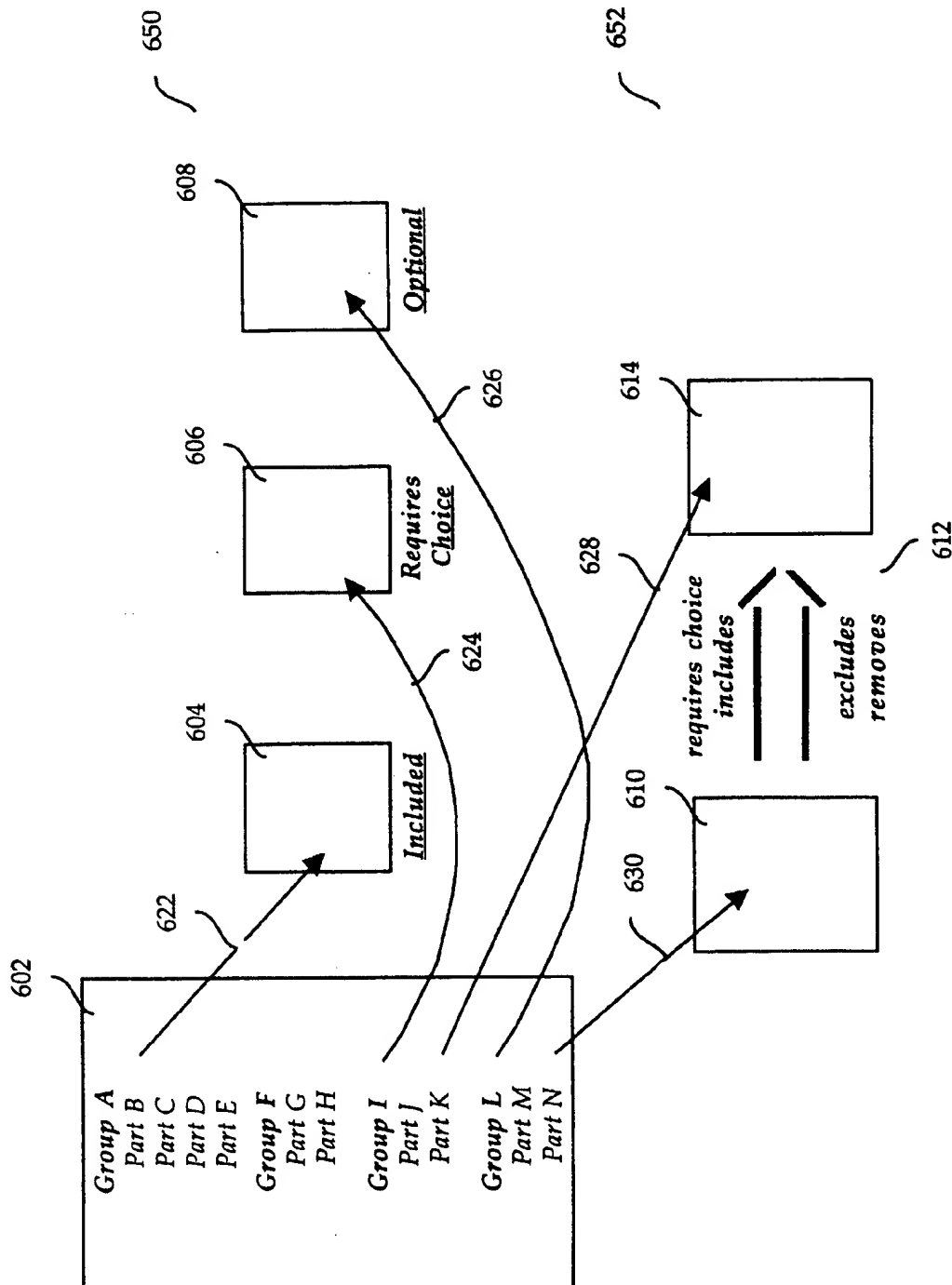


FIG. 6

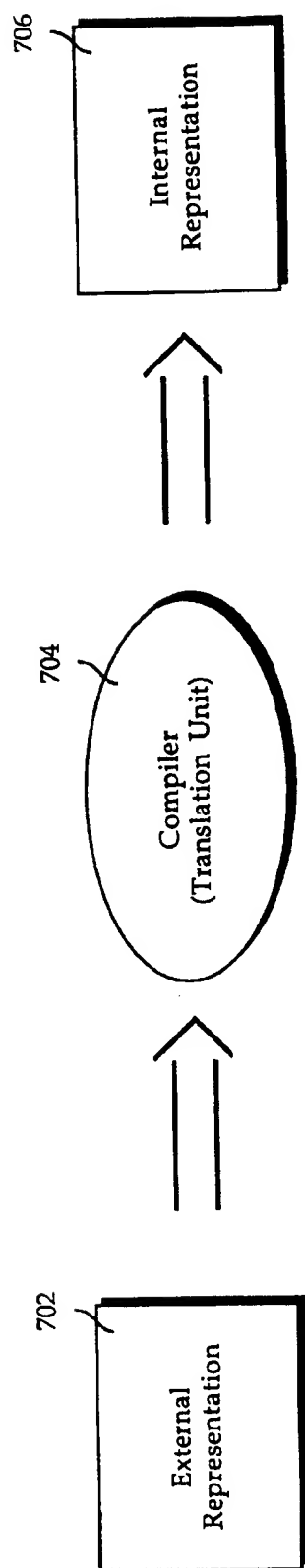


Figure 7

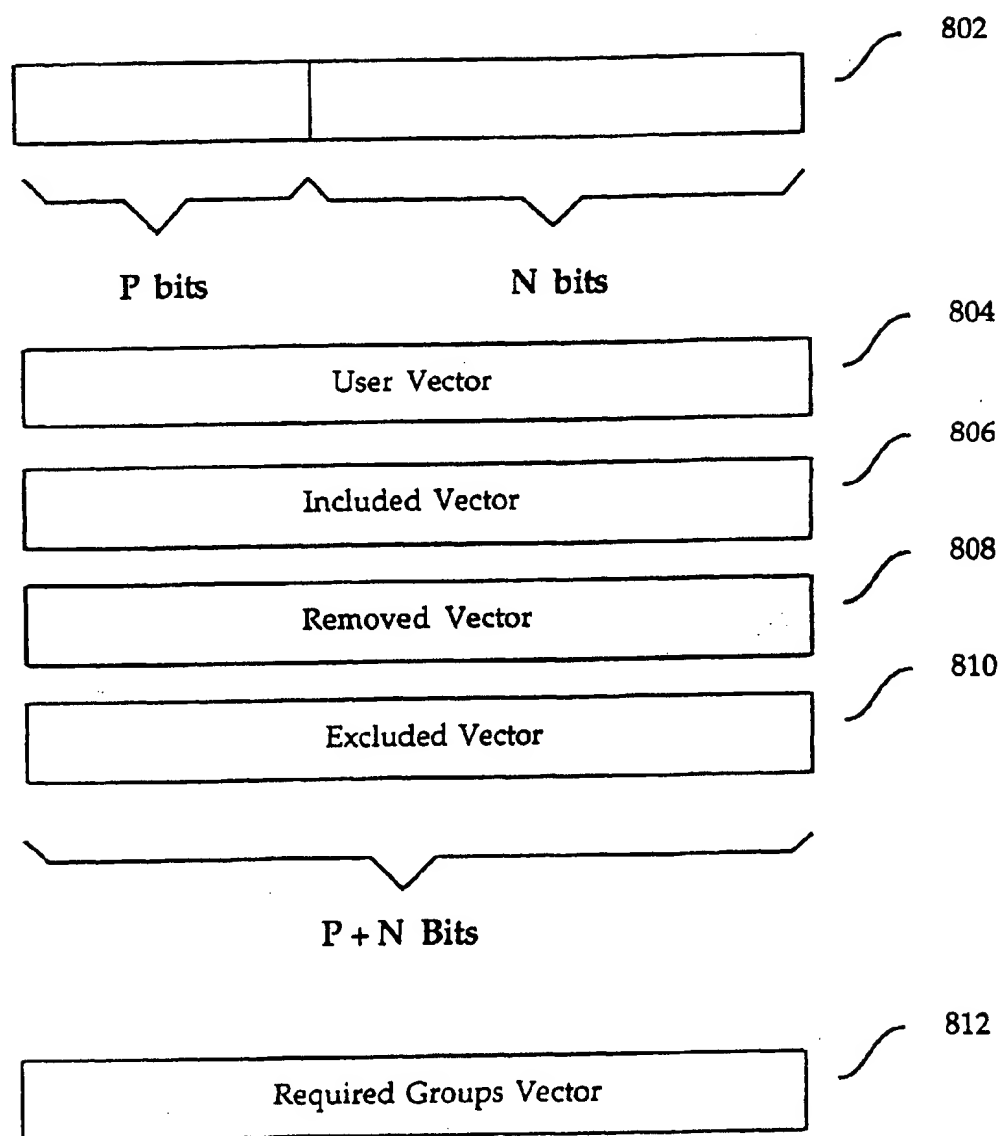


FIG. 8A

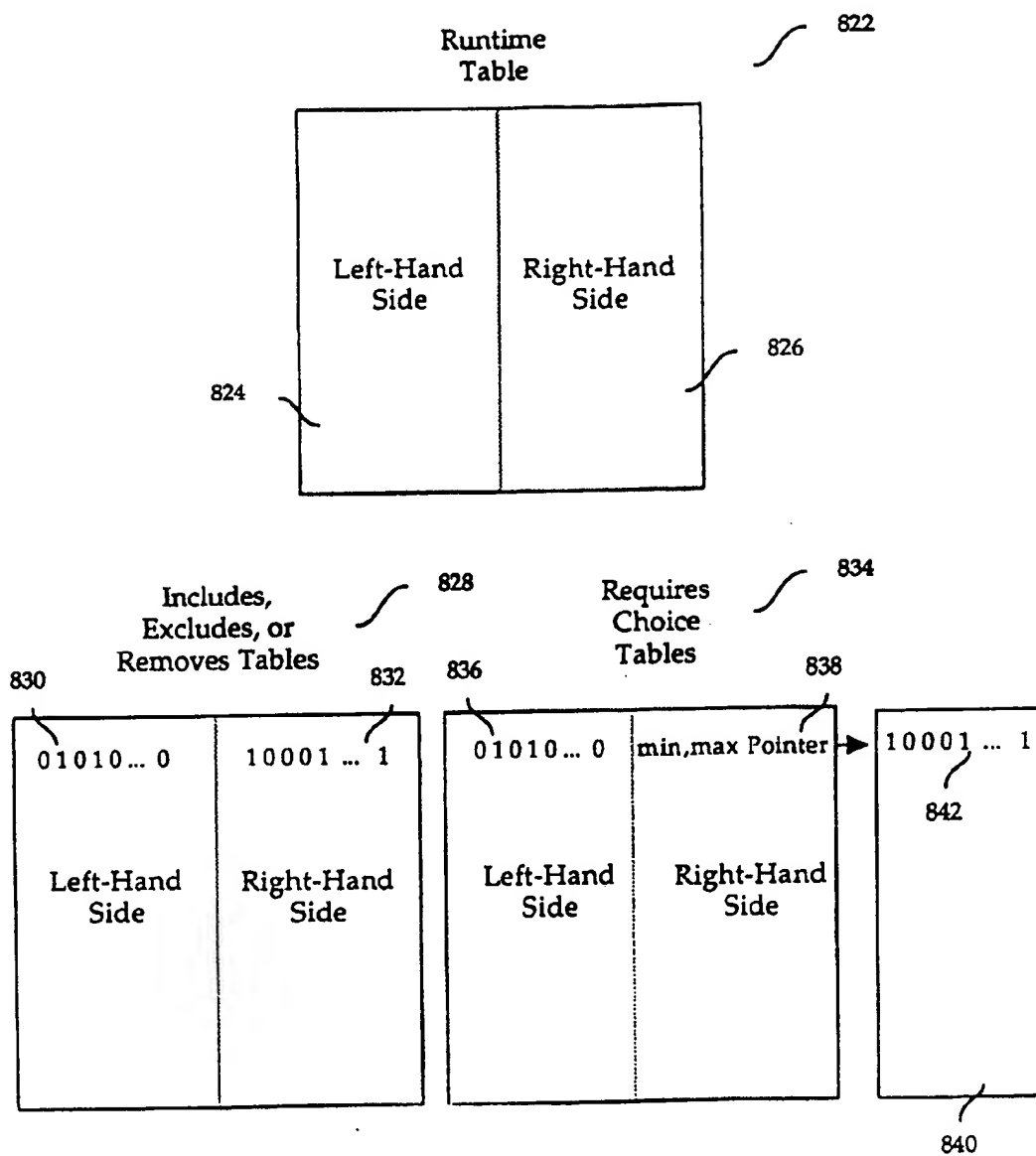


FIG. 8B

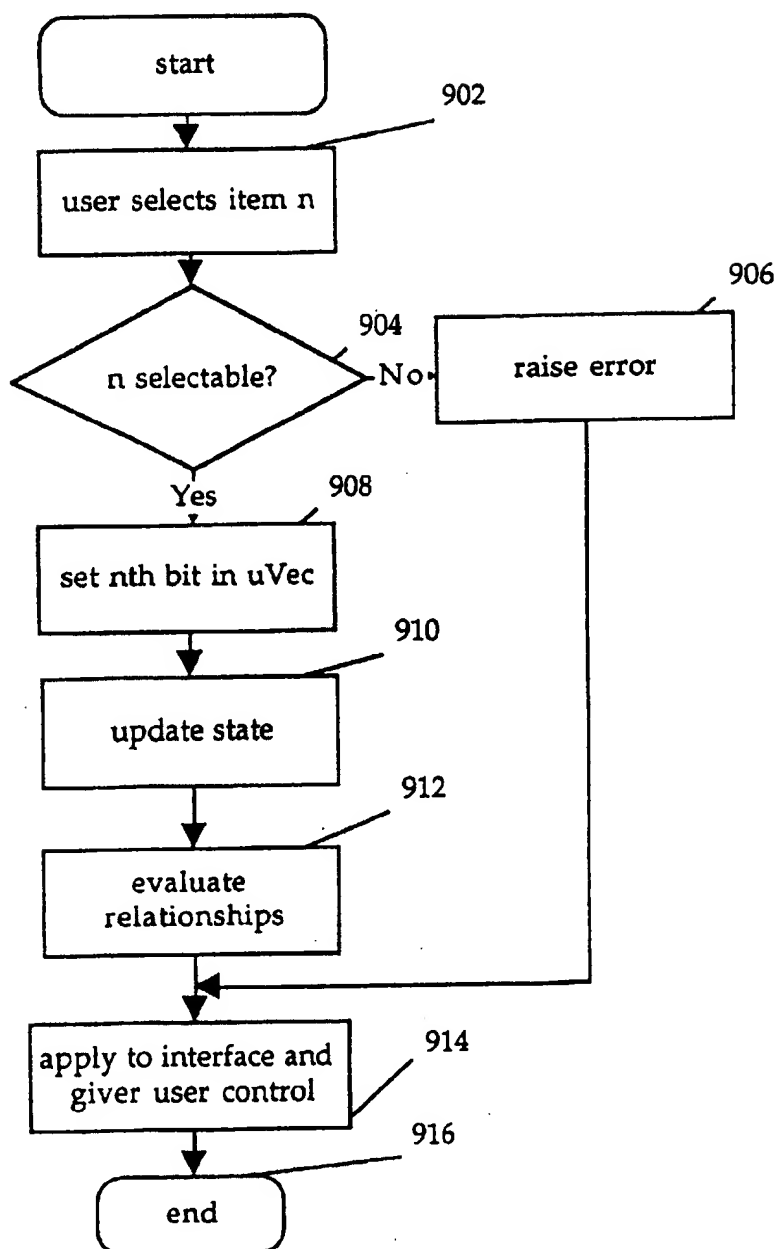


FIG. 9

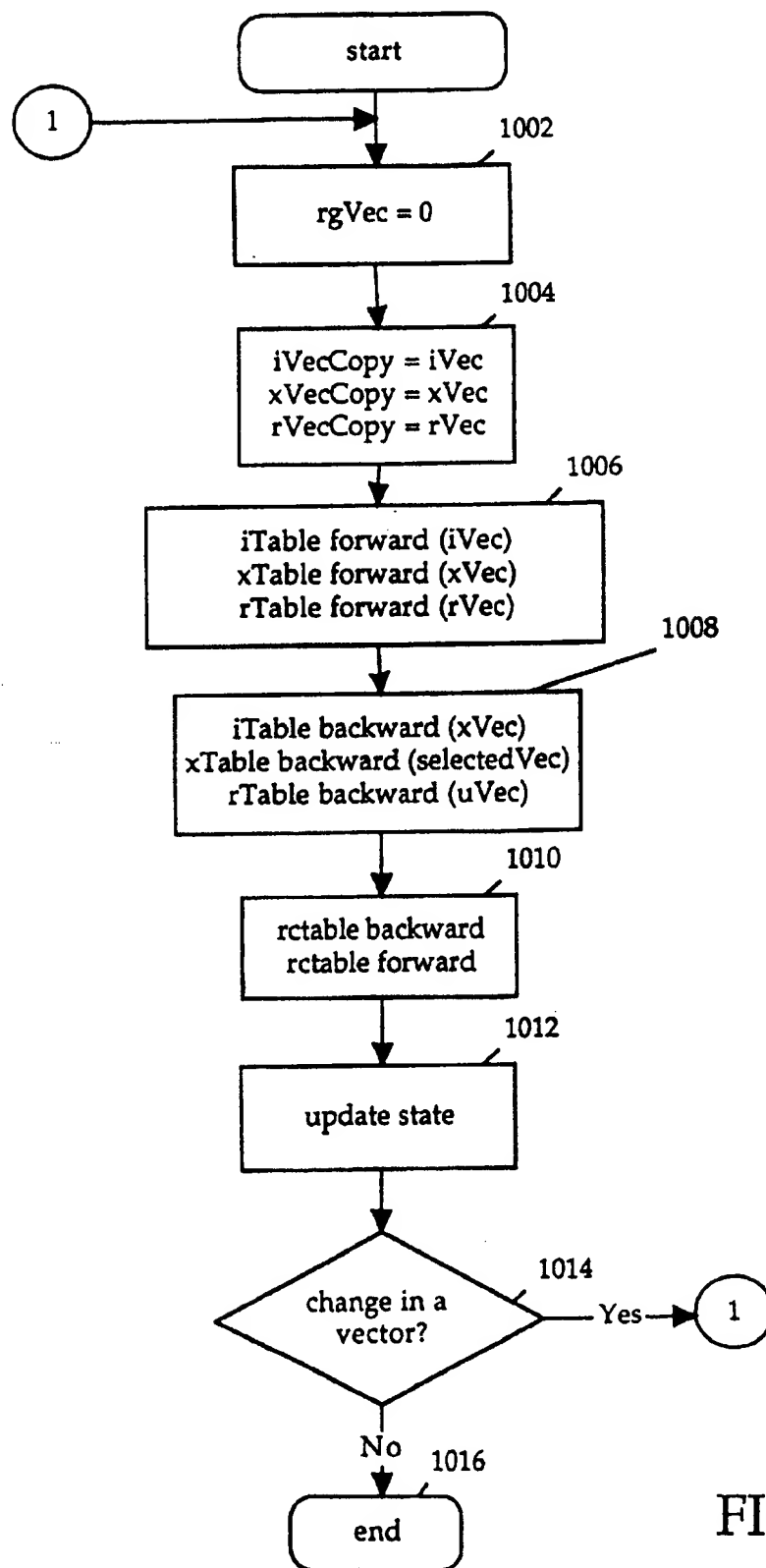


FIG. 10

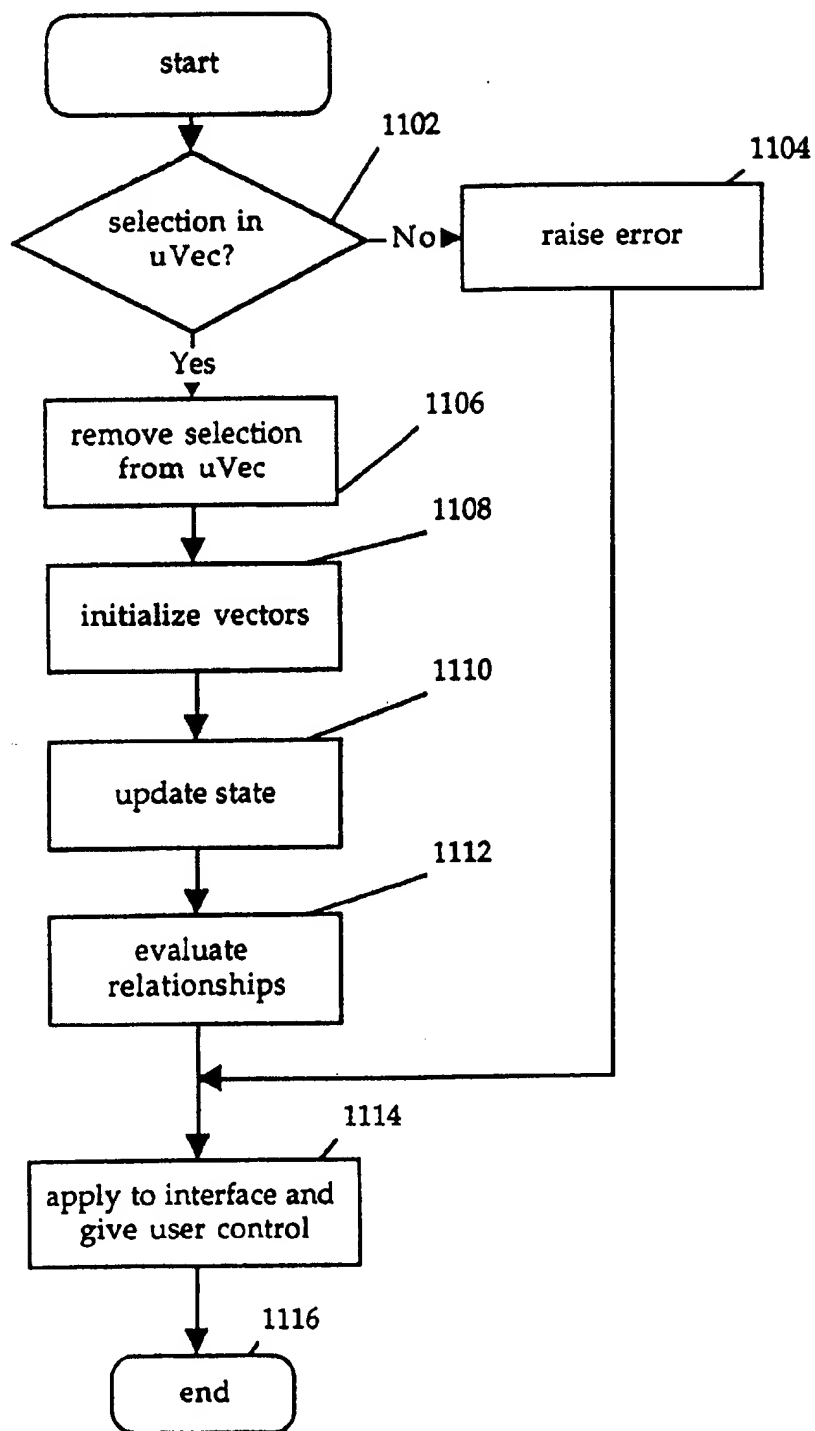


FIG. 11

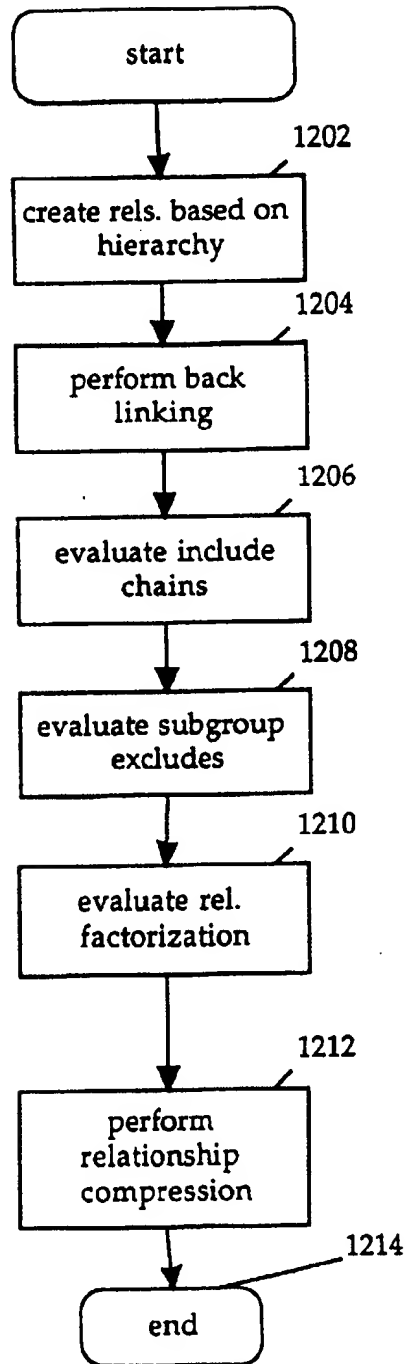


FIG. 12

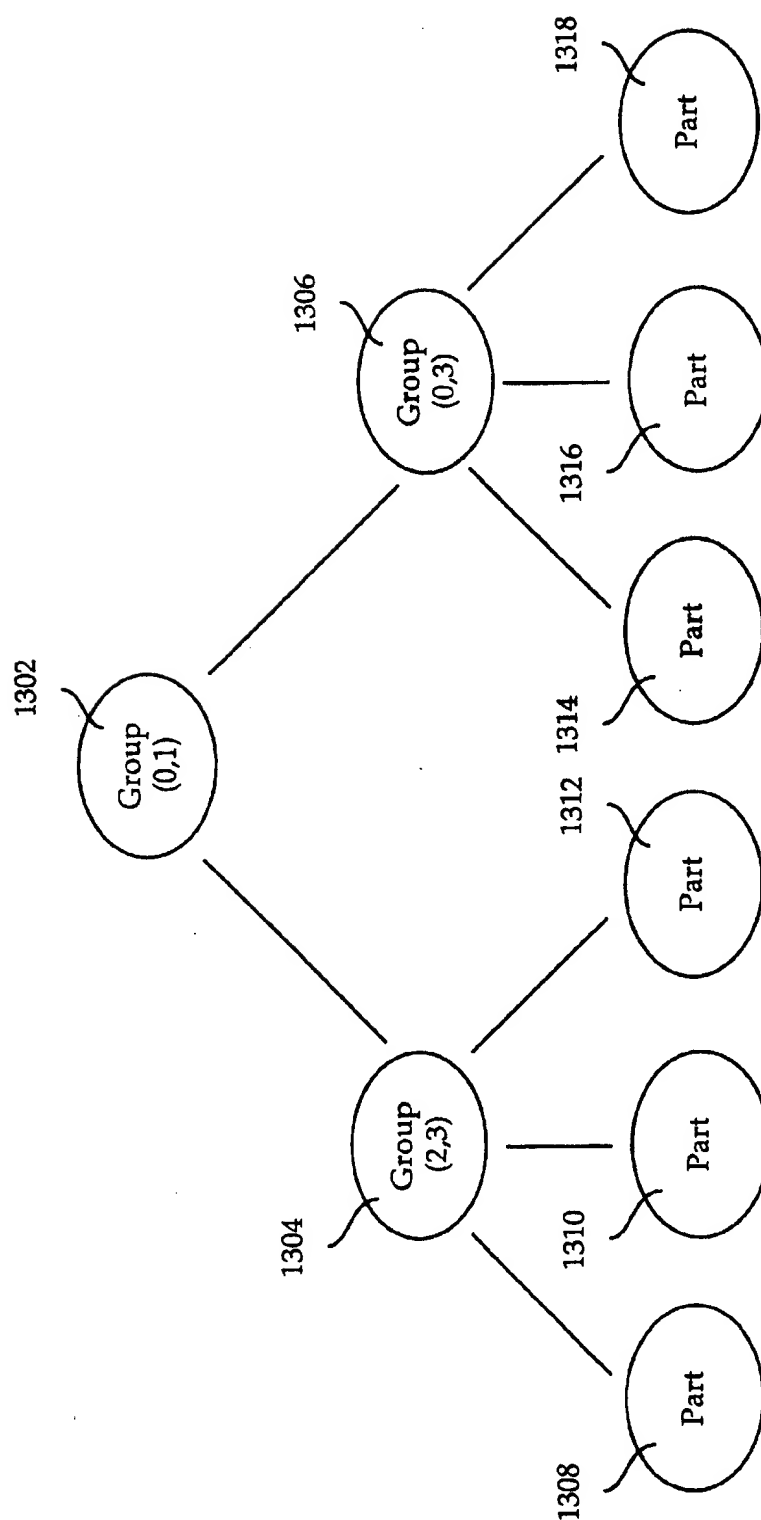


FIG. 13A

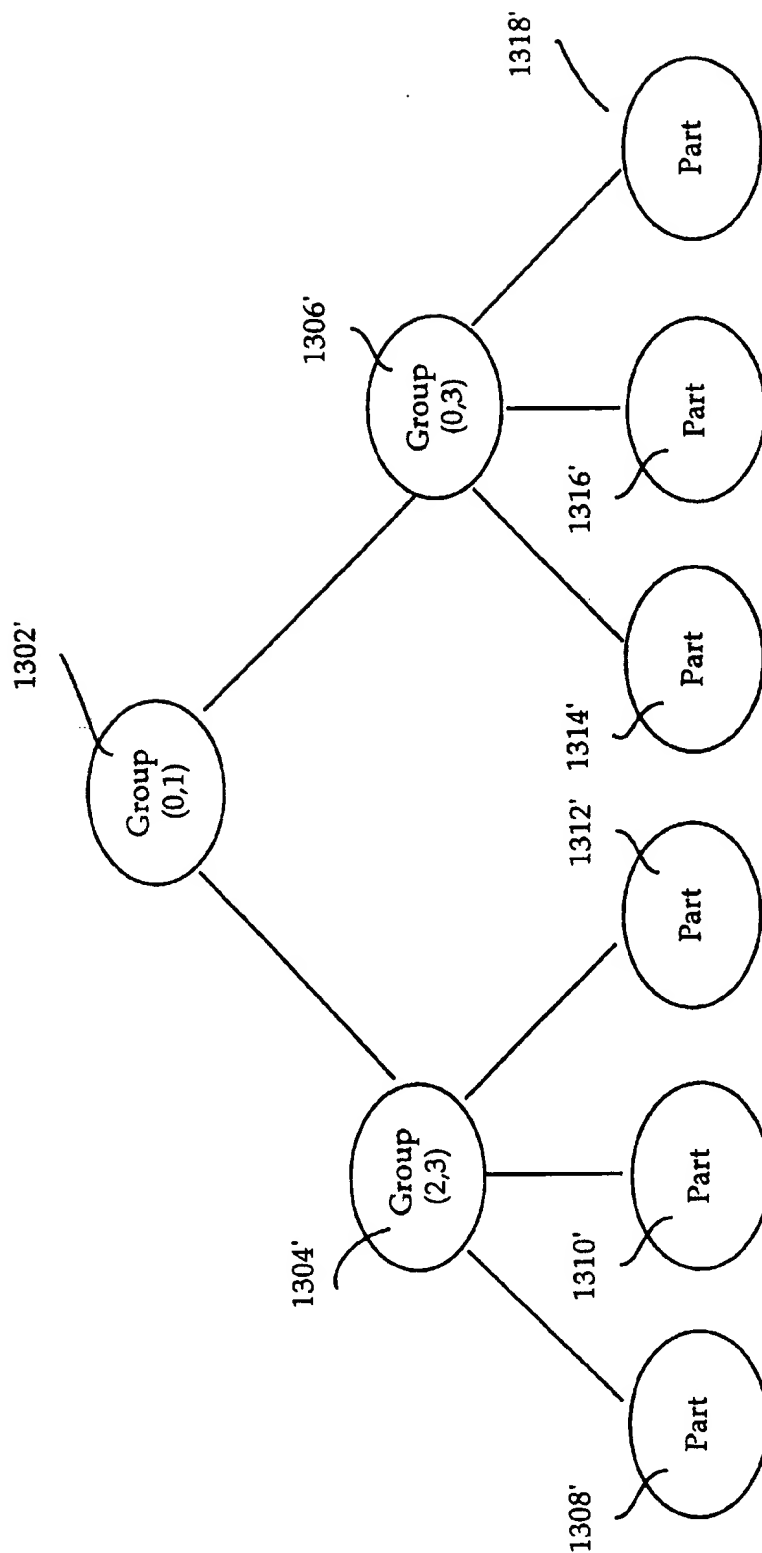


FIG. 13B

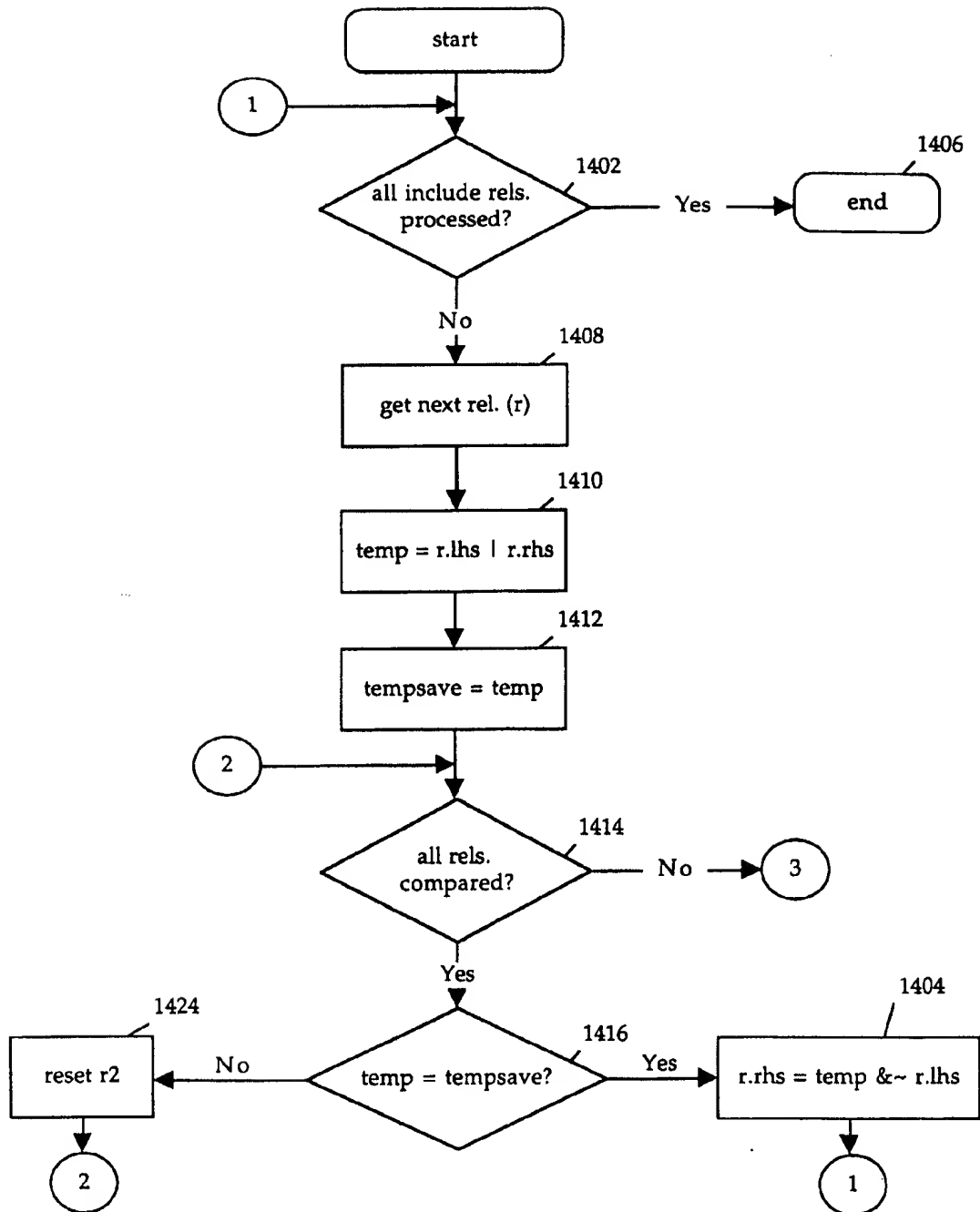


Fig. 14A

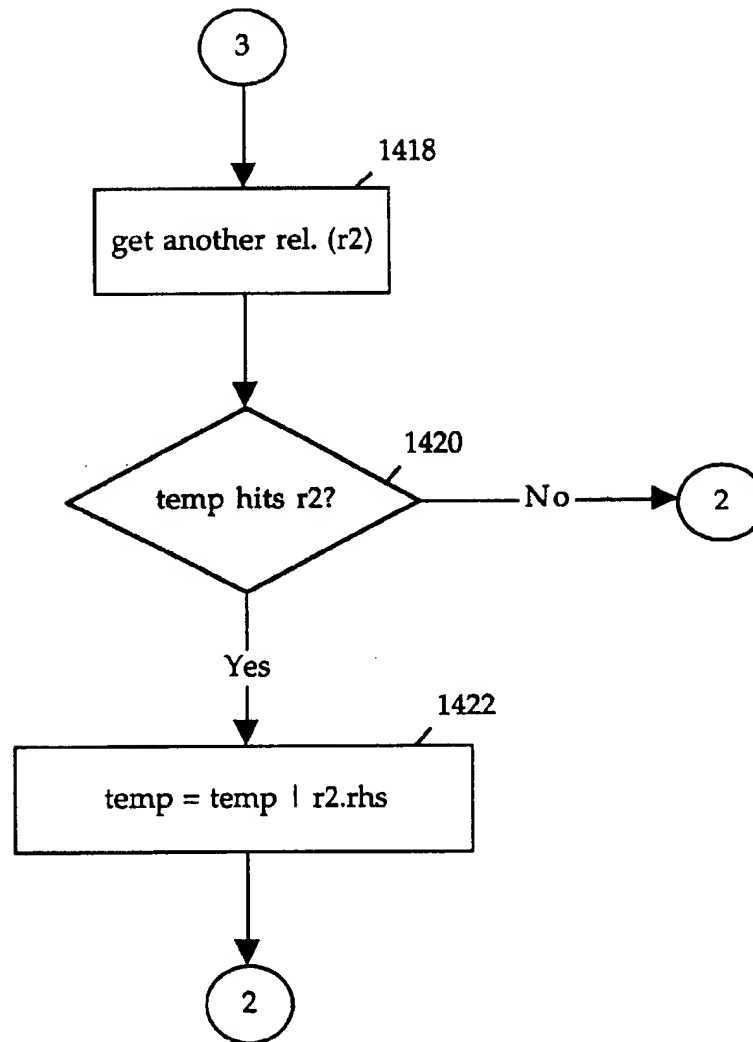


FIG. 14B

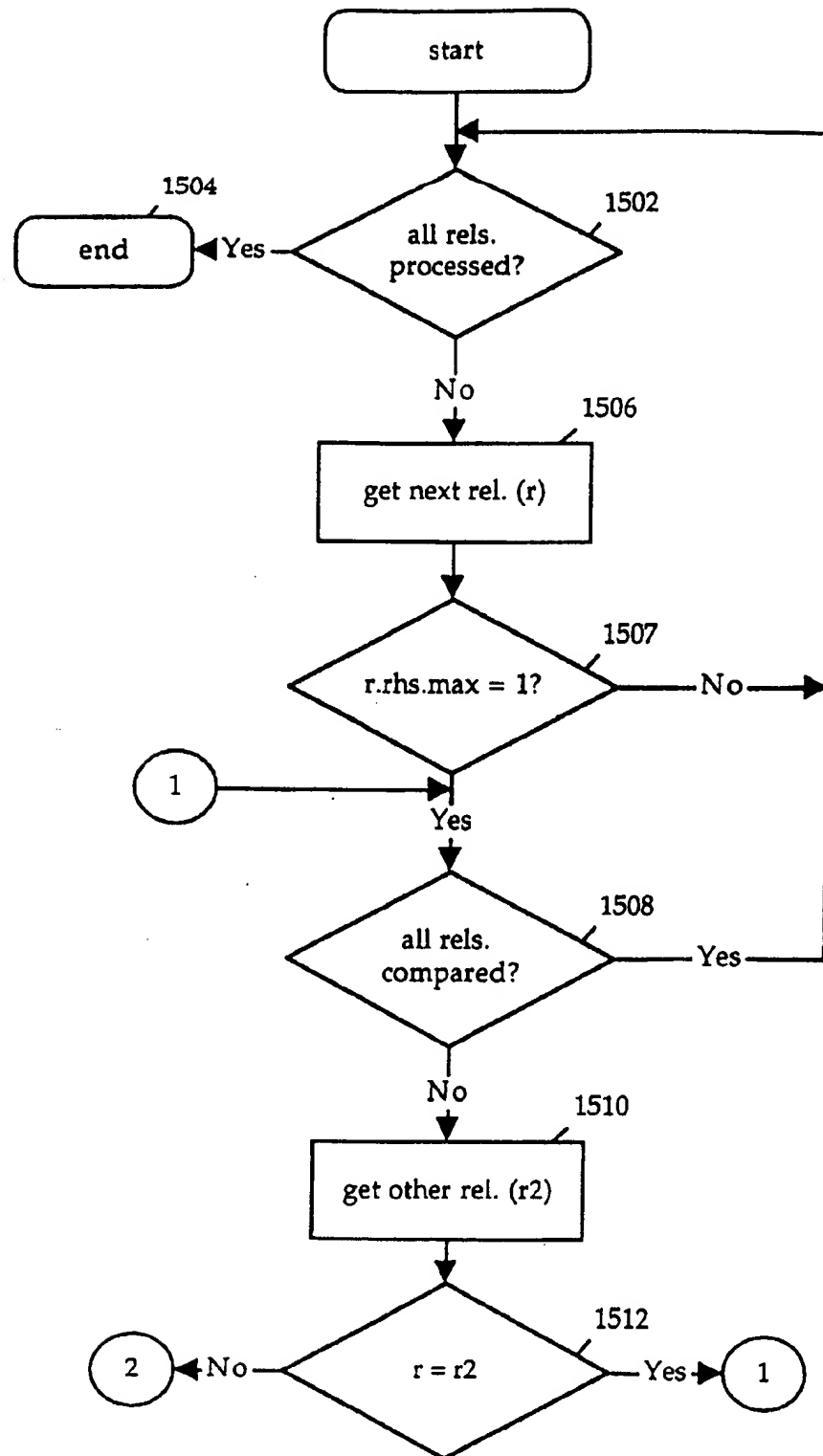


FIG. 15A

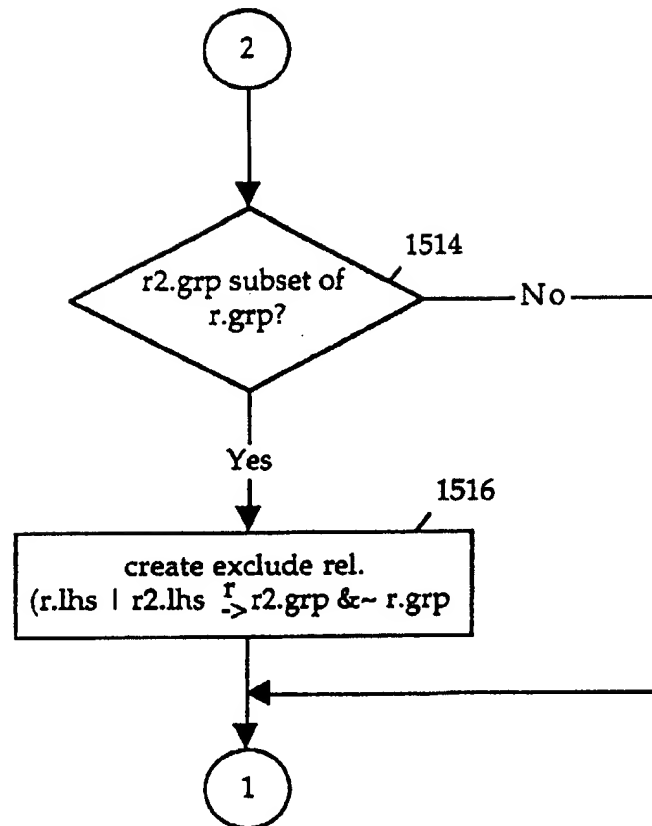


FIG. 15B

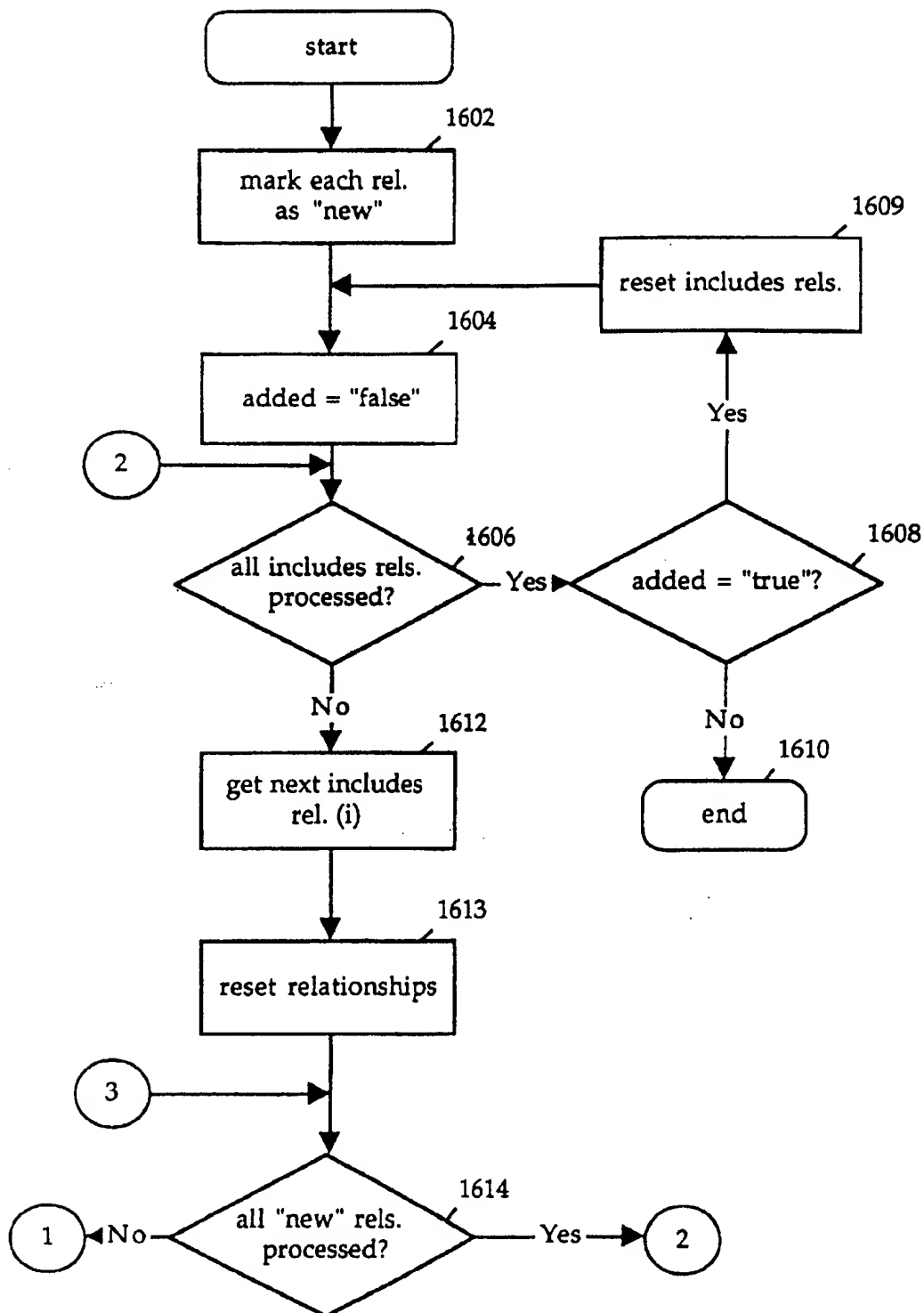


FIG. 16A

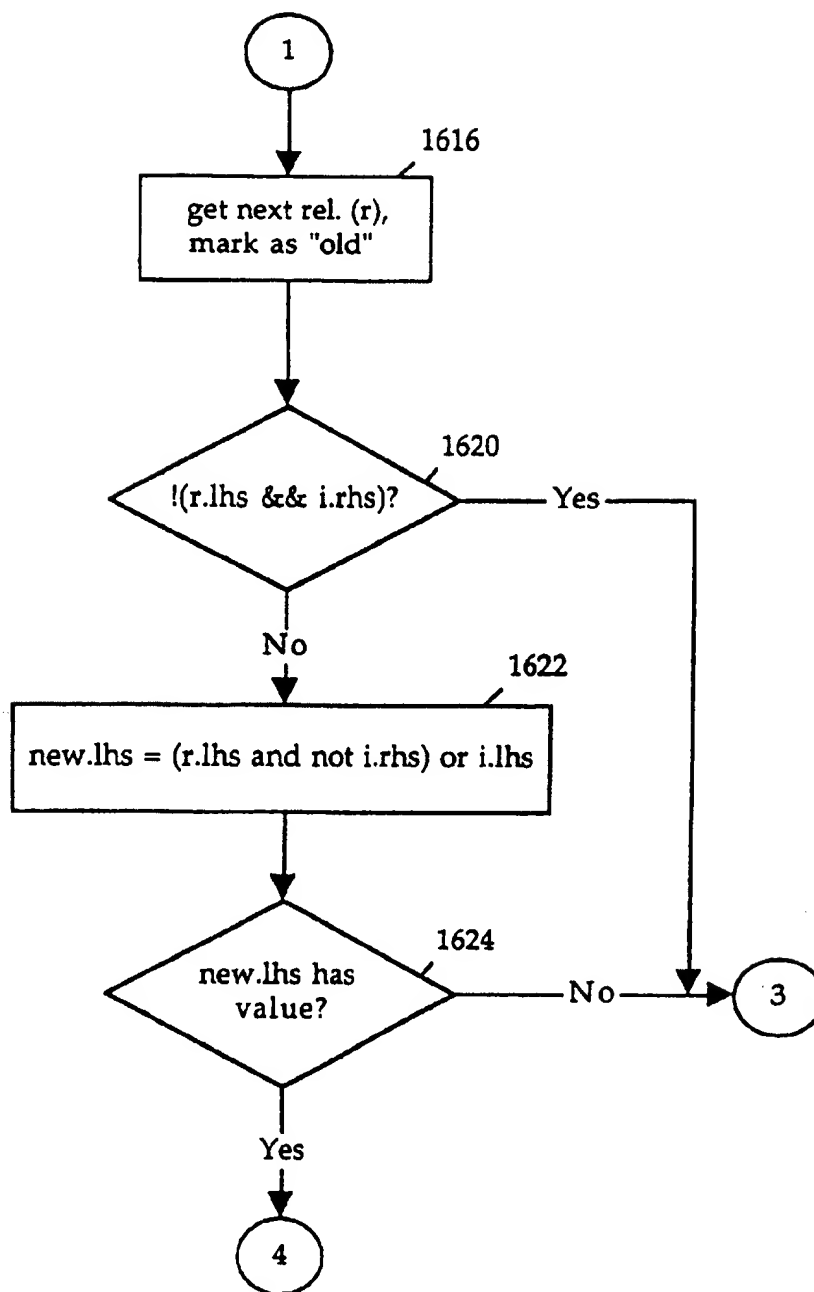


FIG. 16B

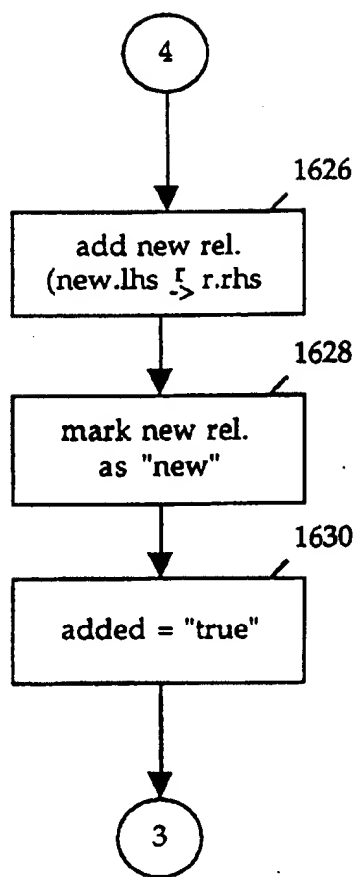


FIG. 16C

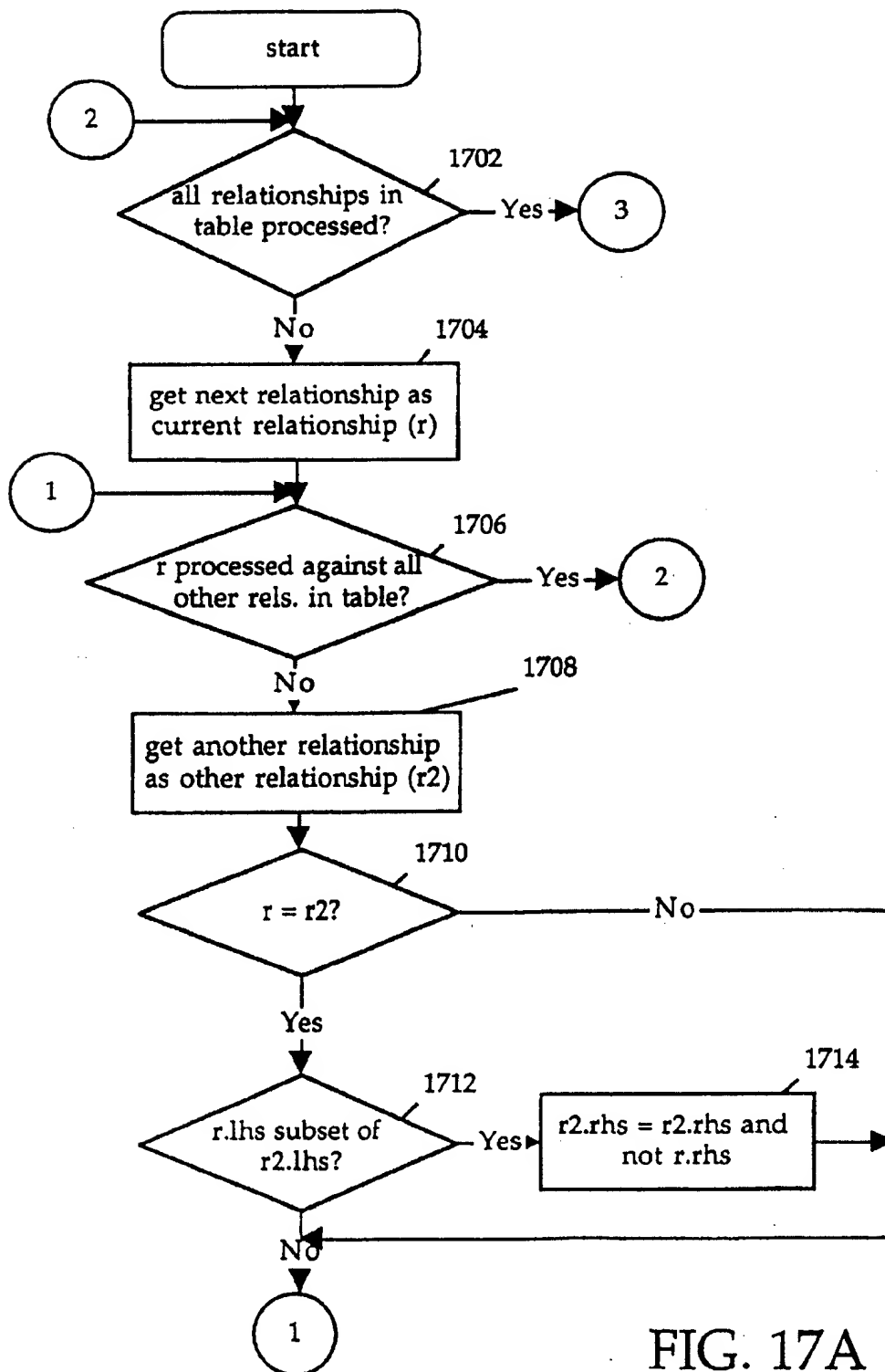


FIG. 17A

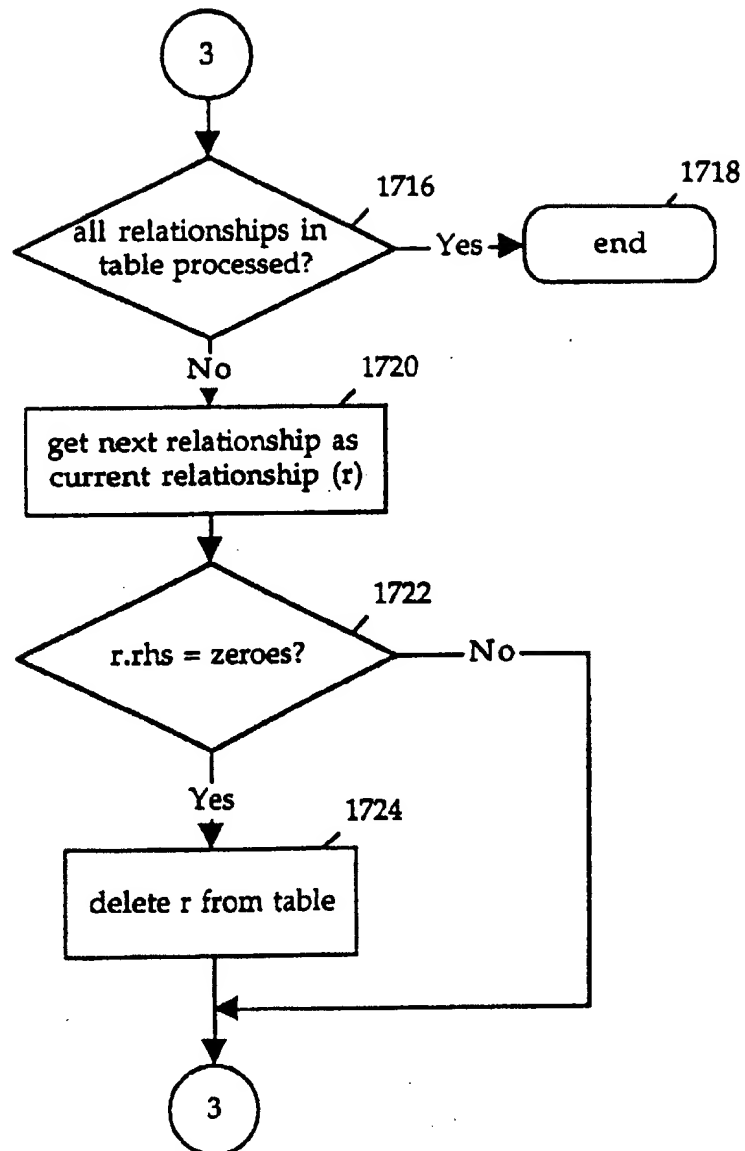


FIG. 17B

METHOD AND APPARATUS FOR MAINTAINING AND CONFIGURING SYSTEMS

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates to maintaining and configuring systems.

2. Background Art

A system is comprised of components. Before a system can be built the components of the system must be identified. To configure a system, a user must select the parts to include in the system. Typically, one who is knowledgeable about a system and its components defines the system. Thus, for example, an automobile salesperson assists an automobile buyer in determining the type and features of the automobile. The salesperson understands the features and options that are available to create a valid configuration. Some features and options cannot be combined. The selection of some features caused other features to be unavailable, etc. It would otherwise be difficult for the buyer to identify all of the features and options available on the automobile that can be combined to create a valid configuration.

Computer systems have been developed to assist one in configuring a system. However, these systems use a configuration language to define a system. Like a programming language, a configuration language uses a syntax that must be understood by a user who is maintaining the data (i.e., a data maintainer). To use one of these configuration systems, it is necessary for a data maintainer to understand the configuration language. This limits the number of users who are able to use the configuration systems. That is, the level of sophistication needed to communicate with the configuration system (through a configuration language) results in less sophisticated users being unable to use the system.

In addition, configuration systems impose a flow or ordering to the user operations. For example, a user is required to remove components from the system in reverse of the order in which they were chosen. Thus, a user may be forced to remove components that the user wants to keep in the configuration to remove an unwanted component. A novice user may have perform many removal operations before achieving an acceptable configuration. If the novice user is required to remove components in a preset order, the user can become frustrated or confused and abort the configuration process.

These systems are designed for a more sophisticated user that has knowledge of the system that is being configured as well as the configuration system used to configure the system. A end user such as an automobile shopper would have difficulty using these systems.

Further, to use these systems a user must be trained to understand the configuration language. Thus, a user who otherwise has knowledge of the systems that are being configured must undergo training to be able to use these configuration systems to configure systems. This leads to increased expenditures such as for training.

SUMMARY OF THE INVENTION

The invention provides the ability to interactively select and configure a product among a set of related products based on availability and compatibility of features and options. It does not impose an order in the selection of products, features or options; only valid selections can be

made at any time. To create an electronic representation of the product information to achieve the above goal, the invention provides a framework for defining a product line.

A product line is defined as a set of related products. A product line has a set of products that contain parts, or components. Parts used to define a product are selected from a parts catalog. Parts in a product definition are related or classified as: included (parts that are included by default), required choices (a choice among a group of parts that must be made to achieve a valid configuration), optional (parts that can be optionally included in the configuration).

Relationships can be defined between the parts in a product definition. A relationship relates a first set of parts with a second set of parts. A set can include multiple parts. The incorporation of parts in a set can be arbitrary. That is, a multi-part set can contain parts that are otherwise unrelated. For example, a set can contain parts such as an engine, sun roof and a color. These parts seem to be unrelated, however, it is possible to combine them into a relationship set for purposes of forming a relationship using the present invention.

Preferably, the part relationships are: included, excluded, removed, and requires choice. An included part is included automatically. A part is excluded from the configuration when its inclusion would result in an invalid configuration. A part may be removed when another part is added. Thus, when a first part exists in the configuration and a second part is added, the first part is removed from the configuration. The requires choice relationship is used to allow a set of choices to be made from a group of parts. The number of parts chosen is limited to a valid bounds specification. The relations that are created between parts within a product are enforced only on that particular product. However, if some part-to-part relationships are to be enforced on all products within a product line, then the relations are generated once and enforced for all products.

A maintenance system is used to define a product. Using the maintenance system, a product can be defined using the product classifications and the part relationships. A graphical user interface (GUI) is used to allow the user to interactively generate a definition. Instead of configuration languages, GUI operations such as drag and drop and selection operations can be used to specify a definition. The notions of included, optional and required choice are easily comprehensible to a user. Further, the idea that parts have interrelationships is also easily understood. Thus, a product can be defined without having to learn a complicated configuration language.

A configuration system is used to configure a system using a definition created by the maintenance system. The configuration system ensures that the current configuration state is always valid. The user can select and unselect parts in any order. When user input is received, the configuration system validates the input based on the current state of the configuration. In addition, the configuration system identifies selections that could cause a valid configuration to become invalid. The configuration removes these selections from the set of possible selections so that the user does not make an invalid selection.

The configuration system evaluates the current state of a configuration based on the product definition, part relationships and state information. After receipt of input from a user, the configuration system evaluates relationships in both the forward and backward direction. Forward and backward evaluations can result in the addition or deletion of elements from the configuration.

The invention uses both an external and internal representation of a definition or definitions. A translation mechanism is used to translate an external representation into an internal representation. The external representation uses a conceptually understandable set of relationships for defining a system and the relationships between the components of the system. The invention takes the definition created by a user and supplements and compresses the definition when necessary to create an internal representation. The internal representation is used during configuration to initialize and validate a configuration based on user input.

During configuration, the invention maintains runtime information that is stored in tables and vectors. To achieve greater processing efficiency, the system represents elements in a configuration (e.g., product, part, and group) as a bit in a bit vector. Thus, for example, a vector has a length that is equal to the total number of elements. An element's bit can be set or reset to specify the state of the element in the current configuration. For example, a user vector can be used that specifies for each element whether the element has been selected by the user during the configuration. In addition, excluded and removed vectors identify whether an element is excluded or removed (respectively) from a configuration. Vectors can be used to identify whether an element 1) has been selected (by the user or the configuration system), 2) is selectable, and 3) notSelectable.

Tables contain element relationships. A table is used to represent the includes, excludes, removes, and requires choice relationships, for example. Each table has a left-hand side and a right-hand side that corresponds to the left-hand and right-hand sides of a relationship. In each case, the left-hand side is a bit vector that contains bits that correspond to elements. The includes, excludes and removes tables contain a bit vector in the right-hand side that represents configuration elements. The right-hand side of the requires choice table is a pointer that points to an entry in a group table. The group table entry is a bit vector that identifies the elements that are contained in the group from which a choice is to be made. The right-hand side of a requires choice table entry further includes minimum and maximum designations. Minimum and maximum values identify the minimum and maximum number of group members that are to be selected to satisfy a requires group relationship.

A bit vector implementation of relationships and internal runtime state allows for fast and efficient computation of relationship based configuration. A comparison of bits can be performed in one machine instruction in most cases.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 provides an illustration of a computer system that can be used with the invention according to an embodiment of the invention.

FIG. 2 provides an overview of the maintenance and configuration systems according to an embodiment of the invention.

FIG. 3 illustrates examples of elements in a parts catalog according to an embodiment of the invention.

FIG. 4 illustrates relationships between parts according to an embodiment of the invention.

FIG. 5 provides an example of product classifications according to an embodiment of the invention.

FIG. 6 provides an example of a GUI screen used in maintenance system 202 according to an embodiment of the present invention.

FIG. 7 provides a block diagram illustrating the conversion process according to one embodiment of the invention.

FIGS. 8A-8B illustrate components of internal representation 706 according to an embodiment of the invention.

FIG. 9 provides a process flow for processing a user selection according to an embodiment of the invention.

FIG. 10 provides an example of a relationship evaluation process flow according to an embodiment of the invention.

FIG. 11 provides an example of an unselect item process flow according to an embodiment of the invention.

FIG. 12 provides a flow for translation processing according to an embodiment of the invention.

FIGS. 13A-13B provide an illustration of groups and nested groups according to an embodiment of the invention.

FIGS. 14A-14B provide an includes chain process flow according to an embodiment of the invention.

FIGS. 15A-15B provide an example of a subgroup excludes process flow according to an embodiment of the invention.

FIGS. 16A-16C provide an example of a relationship factorization process flow according to an embodiment of the invention.

FIGS. 17A-17B provide an illustration of a relationship compression process flow according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

A method and apparatus for maintaining and configuring systems is described. In the following description, numerous specific details are set forth in order to provide a more thorough description of the present invention. It will be apparent, however, to one skilled in the art, that the present invention may be practiced without these specific details. In other instances, well-known features have not been described in detail so as not to obscure the invention.

The present invention can be implemented on a general purpose computer such as illustrated in FIG. 1. A keyboard 110 and mouse 111 are coupled to a bidirectional system bus 118. The keyboard and mouse are for introducing user input to the computer system and communicating that user input to CPU 113. The computer system of FIG. 1 also includes a video memory 114, main memory 115 and mass storage 112, all coupled to bi-directional system bus 118 along with keyboard 110, mouse 111 and CPU 113. The mass storage 112 may include both fixed and removable media, such as magnetic, optical or magnetic optical storage systems or any other available mass storage technology. Bus 118 may contain, for example, 32 address lines for addressing video memory 114 or main memory 115. The system bus 118 also includes, for example, a 32-bit DATA bus for transferring DATA between and among the components, such as CPU 113, main memory 115, video memory 114 and mass storage 112. Alternatively, multiplex DATA/address lines may be used instead of separate DATA and address lines.

In the preferred embodiment of this invention, the CPU 113 is a 32-bit microprocessor manufactured by Motorola, such as the 680X0 processor or a microprocessor manufactured by Intel, such as the 80X86, or Pentium processor. However, any other suitable microprocessor or microcomputer may be utilized. Main memory 115 is comprised of dynamic random access memory (DRAM). Video memory 114 is a dual-ported video random access memory. One port of the video memory 114 is coupled to video amplifier 116. The video amplifier 116 is used to drive the cathode ray tube

(CRT) raster monitor 117. Video amplifier 116 is well known in the art and may be implemented by any suitable means. This circuitry converts pixel DATA stored in video memory 114 to a raster signal suitable for use by monitor 117. Monitor 117 is a type of monitor suitable for displaying graphic images.

The computer system described above is for purposes of example only. The present invention may be implemented in any type of computer system or programming or processing environment.

The invention maintains and configures systems. The invention eliminates the need for a user to learn a configuration language or write code to maintain and/or configure a system. A user interface uses various operations such as drag and drop and item selection to define a product, for example. Elements that comprise a definition (e.g., of a product) can be added or removed in any order. No order is imposed on the user. There is no requirement, for example, that the user remove parts of a product in the order in which they were added. No fixed flow or order is required to edit a given product.

A definition includes an identification of the components that comprise the definition and the interrelationships between the components. The interrelationships are conceptually easy for the user to understand. The same relationships are used to define and configure any system. An external representation that includes these relationships allows a user to view and maintain the definition. The external representation is translated into an internal representation that is designed to decrease processing time and increase response time.

In addition, during a configuration session, the invention is capable of allowing a user to select or unselect configuration items without imposing a flow on the user. There is no order imposed on the user in terms of the sequence in which items are selected for the configuration or unselected from the configuration. For example, it is not necessary to select a product before choosing options. The invention can identify the products that are still available based on the options that have already been selected. Further, a user can unselect an item (e.g., delete the item from the configuration) without regard to the order in which it was selected (e.g., added to the configuration).

Examples of systems that can be maintained or configured using the invention include automobiles, computers, time clock machines, and shoes. Terms such as part, product line, parts catalog, and product are used herein for illustration purposes. It should be apparent that this invention can be used to configure systems that are not limited to products and product lines, etc.

FIG. 2 provides an overview of the maintenance and configuration systems according to an embodiment of the invention. Maintenance system 202 maintains a parts catalog 204, parts relationships 206 and product definitions 208. Maintenance system 202 uses a user interface that includes the ability to add items to parts catalog 204 and specify part relationships 206 and product definitions 208. The user interface displays a set of hierarchies that provide a conceptually easier way of viewing a definition. The invention maps the set of hierarchies (an external representation) to an internal representation.

The internal representation is used by configuration system 212 to maintain and configure systems based on user input. Configuration system 212 provides the ability to specify a product. Configuration system 212 verifies the product specification. Using configuration system 212, a

user can interactively select and configure a product among a set of related products based on availability and compatibility of features and options. It does not impose an order in the selection of products, features or options. The configuration system 212 allows the user to only make valid selections. However, selections can be made in any order.

Parts catalog 204 consists of parts that are components of products. Similar parts are grouped together to form a part hierarchy. Easy maintenance of relationships is achieved by the hierarchy. For example, when a group of parts is assigned a behavior, all the members inherit that behavior automatically. FIG. 3 illustrates examples of elements in a parts catalog according to an embodiment of the invention. Referring to FIG. 3, a parts catalog (e.g., parts catalog 204) contains parts 302-310, 316, and 318. A parts catalog can also contain a group of parts such as group 312. Group 312 contains parts 306-310. A group can contain other groups. For example, group 314 contains group 312 and parts 316-318. Behavior assigned to elements of group 314 is inherited by members of group 312.

Part-to-part relationships can be created between parts within a product. Relationships are defined between a first set of parts and a second set of parts. During configuration, when all of the members of the first set of parts are selected, the relationship between the two sets is enforced on the parts in the second set. In the preferred embodiment, there are four kinds of relationships between parts: requires choice, includes, can't work with (or excluded), and removes. FIG. 4 illustrates relationships between parts according to an embodiment of the invention.

Part 402 includes part 404. The includes relation causes a set of parts in a second set (e.g., part 404) to be included in the configuration when a first set of parts (e.g., part 402) is selected. For example, a luxury package includes a CD player; when the luxury package is selected, a CD player is included in the configuration. The can't work with (or excluded) relation ensures that a set of parts from a second set are never in the same configuration as parts in the first set. For example, part 402 (e.g., sun roof) can't work with part 406 (e.g., a roof-top antenna). When part 402 is selected, part 406 cannot be selected. Part 406 is excluded such that it cannot be selected.

When part 402 is selected, part 414 is removed from the configuration. The removes relation causes items that are included in a second set of to be removed from the configuration when the first set is selected. For example, when a high end stereo is selected the standard stereo is removed from the configuration.

The requires choice relation recognizes that a choice (between a minimum and maximum number) has to be made between a second set of parts (or members of a group) to ensure a valid configuration when the parts in a first set are selected. Part 406 (e.g., climate control feature) requires that a choice be made among parts 408-410 (e.g., 1 zone A/C or 2 zone A/C). A requires choice relationship requires that a number of items be selected based on minimum and maximum values from the right-hand side of the relationship to satisfy the relationship.

Parts 408 and 410 can be combined to form group 416. Group 416 can be defined by the user. If the user does not define group 416, the invention preferably creates group 416 to contain parts 408 and 410. Thus, when two or more parts are defined on the right-hand side of a requires choice relationship, the invention preferably creates a group that contains these parts. The requires choice then becomes a requires choice on the group (e.g., group 416).

Parts are used to illustrate relationships in FIG. 4. However, a relationship can contain groups. To illustrate, a group of parts can be substituted wherever a part is used in the illustration. Thus, for example, parts 402, 404, 406, 408, 410, and 414 can each be replaced by a group of parts. Thus, for example, if parts 402 and 404 are replaced by groups (e.g., groups 402 and 404, respectively), when the members of group 402 are selected, the members in group 406 are included in the configuration. To further illustrate, part 408 can be replaced by group 408. In that instance, group 408 is a member of group 416. A hierarchy of groups (e.g., group 408 contained within group 416) can be used in a relationship.

Parts can be combined to form an arbitrary grouping. Thus, it is not necessary to combine parts in a group based on a logical or intuitive relationship between the parts in the group. For example, a group can contain an engine, a color, and a sun roof.

The relations that are created between parts within a product are enforced only on that particular product. However, if some part-to-part relationships are to be enforced on all products within a product line, then the relations are created once and are enforced for all products. These relationships are referred to as global relationships.

A product includes zero or more elements of parts catalog 204. Product definition 208 (see FIG. 2) is generated by populating it with its component parts. Product definition 208 is generated by population of a product with its component parts. The parts within a product are classified as one of three different types: included parts, required choices, or optional parts. A part that is not classified as one of these types is assumed to be unavailable in that product. FIG. 5 provides an example of product classifications according to an embodiment of the invention.

An included part is a part that is included in a product by default. For example, parts 504 and 506 are automatically included in product 502. For example, when a configuration user chooses the product definition product 502, the parts 504 and 506 are automatically included in the configuration. A required choices classification specifies that a choice among a group of parts has to be made to create a valid product configuration. For example, product 502 (e.g., automobile) can include a color group 518 containing red (part 508), green (part 510) and blue (part 512) that is a required choice. In configuring a product, the user must choose a color in this group to create a valid product configuration. Parts 514 and 516 are optional parts. An optional part is not required for a valid configuration.

A product line is defined as a set of related products. The invention provides the framework for defining a product line. To define a product line: all parts (components) in the product line are entered into parts catalog 204. An instance in products definition 208 is created that identifies parts from parts catalog 204 that are assigned to a product. Part relationships 206 can also be defined for the product. Maintenance System

Parts catalog 204, part relationships 206, and product definition 208 are created and maintained using maintenance system 202. While any method can be used for creation and maintenance, a graphical user interface (GUI) is preferably used. A text editor could also be used for data entry, for example. FIG. 6 provides an example of a GUI screen used in maintenance system 202 according to an embodiment of the present invention. One who uses the GUI screen (a maintainer) can define a product (e.g., an automobile). A product definition is used by a configuration user to config-

ure a system defined in maintenance system 202. Parts catalog 204 uses a database or other type of storage and retrieval capability, for example, to store information about its parts (e.g., part number, description, price, etc.).

The GUI screen of FIG. 6 is divided into a product definition section 650 and a part relationship definition section 652. Pane 602 displays elements from parts catalog 204. Panes 604, 606, 608 are used to define a product. Panes 604, 606, and 608 represent classifications or product relationships. Panes 610 and 614 and relationship 612 are used to define part relationships.

A user can drag elements from pane 602 to panes 604-608 to define a product. For example, to include Part B in the product definition, Part B is dragged from pane 602 to pane 604. Alternatively, to drag parts B, C, D, and E, group A can be dragged from pane 602 to pane 604. Group A and its component parts (parts B, C, D, and E) are thereby included in the product definition. Similarly, a user can specify that a configuration user must choose a part from a group, e.g., Group I, by dragging one or more parts or a group into pane 606. An optional part or group can be identified by dragging an element, e.g., Group L, into pane 608. If an element from pane 602 is not moved to one of panes 604-608 it is assumed that the maintainer wants to exclude that element from the product that is being defined. The product-level relationships or classifications (or types) illustrated in FIG. 5 can be defined using panes 604-608.

A subset of the product-level relationships map to a subset of the part relationships. For example, the include and requires choice product-level relationships map to the same named part relationships (as defined in FIG. 4 and shown in relationships 612). Minimum and maximum values set for a requires choice product-level relationship map to the minimum and maximum values of a requires choice part relationship. Elements that are not dragged into one of panes 604, 606 or 608 are considered to be excluded.

In addition to defining the elements (and their types) that comprise the product, the maintainer can specify relationships between parts of the product. Panes 610 and 614 and relationship 612 are used to define relationships between parts. The maintainer can drag an element (or elements) from pane 602 into pane 610. For example, the user can drag Part N from pane 602 into pane 610. The element(s) dragged into pane 610 are referred to as the left-hand side of the relationship. The element (or elements) from pane 602 that is related to the left-hand element(s) is dragged into pane 614. The element(s) dragged into pane 614 are referred to as the right-hand side of the relationship. For example, the user can drag Part K from pane 602 to pane 614.

To identify the type of relationship that exists between the left-hand side and right-hand side elements, the maintainer chooses one of the four part-to-part relationships from relationship 612. For example, if the maintainer chooses the includes relationship, the right-hand side elements are included in the product when a configuration user selects the left-hand side element(s). Thus, for example, Part N includes Part K. When Part N is chosen by a configuration user, Part K is automatically included in the configuration.

When the maintainer completes the product definition using the GUI screen, the product definition is saved (e.g., in parts definition 208). The product definition can then be used by a configuration user in configuration system 212. External and Internal Representations

The GUI screen of FIG. 6 illustrates a view of a product as seen by a user (e.g., maintainer and configuration user). A view of the product that is seen by a user is referred to as the external representation. A user sees a product or product

definition from a conceptual level that is easy to understand. The user does not need to express a definition or configuration using a language that has a syntax that is foreign to the user. A GUI having graphical selections and operations is preferably used to allow the user to see the external representation. The external representation uses terminology that is familiar or intuitive to the maintainer. For example, an include product relationship defines elements that can be included in a product.

An external representation seen by the user is translated into an internal representation that can be used by the invention to process user input. FIG. 7 provides a block diagram illustrating the conversion process according to one embodiment of the invention. Compiler 704 is used to perform the conversion. Compiler 704 generates internal representation 706 from external representation 702.

External representation 702 is the representation of a product definition that is created using the maintenance system of the invention. External representation 702 provides a graphical representation of a system definition, its components and their interrelationships. FIG. 6 can be used to display external representation 702. External representation 702 can include a multi-level hierarchy of parts that help to define the product at a conceptual level. The hierarchies are not required to drive a configuration session, however. Therefore, compiler 704 flattens out the hierarchies. A set of relationships is generated using external representation 702 during a translation process.

The set of relationships generated for a system definition by compiler 704 comprise internal representation 706. Internal representation 706 is generated from external representation 702 and can be used to control a configuration session. In addition to internal representation 706, a configuration state is used during the configuration session. FIGS. 8A-8B illustrate components of internal representation 706 and the configuration state according to an embodiment of the invention.

Referring to FIG. 8A, vector 802 provides an example of a vector used to store state information. Vector 802 is comprised of bits. A set of p bits is used to represent the products that are known to maintenance system 202 and configuration system 212. Each bit of the P bits corresponds to a product. A set of N bits is used to represent the parts. Each bit in the N -bit set corresponds to a part in parts catalog 204, for example. Vector 802 is therefore $P+N$ bits in length.

Vectors 804-810 provides implementation examples of vector 802. Vectors 804-810 are maintained by configuration system 212 in a configuration session. Vector 804 is a user vector that contains the set of elements that are selected by the user. Vector 804 is referred to as the user vector, or $uVec$. The set of elements that are included in the configuration by the configuration system (e.g., an item that is automatically included when the user selects the left-hand in an includes relationship) is contained in included vector 806, or $iVec$. Removed vector 808, or $rVec$, contains the elements that are removed from the configuration by configuration system 212. Elements excluded from the configuration by configuration system 212 are identified in excluded vector 810, or $xVec$.

Each element that can be used to configure a system has a bit in these vectors. When an element is included, excluded, removed (or deleted), or selected, the corresponding bit in the appropriate vector ($iVec$, $xVec$, $rVec$, or $uVec$) is set. The use of vectors to represent elements results in increased efficiency in processing. Bit-wise operations can then be used during processing. For example, it is possible to determine whether a group of elements have been selected

by the user by performing a bit comparison using the $uVec$. The increased efficiency results in optimal response times that are necessary for an interactive system such as is preferred for configuration system 212. The bit vector implementation used by the invention allows for fast and efficient computation of runtime algorithms. A bit vector implementation of relationships and internal runtime state allows for fast and efficient computation of relationship based configuration.

Another vector that contains state information is required groups vector 812, or $rgVec$. The $rgVec$ contains a bit for each of the required groups specified for a product. As previously discussed, the maintainer can include a required group in a product definition by dragging a group from pane 602 to pane 606, for example (see FIG. 6). The number of bits in $rgVec$ can differ from the number of bits in vectors 802-810. The number of bits contained in $rgVec$ corresponds to the total number of required groups defined for a product. Each bit in $rgVec$ is set to indicate that it has not yet been satisfied. It is unset (set to zero) when a required group has been satisfied. Thus, when a user attempts to accept a configuration, configuration system 212 can determine whether all of the required choices have been satisfied.

In addition to these vectors, additional vectors are used to determine a current state. The current state is defined by three types of state information that track the state of parts in the configuration. The current state is visible to the user. The three state types are: selected, selectable, and notSelectable. The selected state identifies all of the parts that are currently included in the configuration. The selectable state identifies the parts that are not in the configuration but that can be selected by the user for the configuration. The notSelectable state identifies the parts that are not in the configuration and cannot be selected by the user. Configuration system 212 ensures that only parts that can form a valid configuration are selectable; those that cannot are notSelectable.

The $uVec$, $iVec$, $xVec$, and $rVec$ vectors can be used to generate the three visible states. The states are dynamic and can change based on user input. The following provides definitions using the formulae that can be used to generate the three visible states (+is union and -is difference):

$$\text{Selected} = (uVec + iVec) - rVec$$

$$\text{NotSelectable} = xVec + rVec$$

$$\text{Selectable} = [\text{AllItems}] - \text{Selected} - \text{NotSelectable}$$

Computation of the visible state using C++, for example, can be accomplished using the following C++ code (the syntax assumes appropriate bit-wise operator overloads for the given operations):

$$\text{SelectedVec} = (uVec | iVec) \& \sim rVec$$

$$\text{NotSelectableVec} = xVec | rVec$$

$$\text{SelectableVec} = \sim (\text{SelectedVec} | \text{NotSelectableVec})$$

The selected state is the union of the elements selected by the user or automatically included in the configuration minus the elements removed by configuration system 212. The notSelectable state is the union of the elements that are excluded or removed. The selectable state is the set of all elements minus the elements already selected and the elements that cannot be selected. The selected, selectable, and notSelectable states can be generated from these four vectors when needed. They can be stored as vectors that have the structure of vector 802.

Runtime tables are preferably used to store internal representation 706 for a configuration session. FIG. 8B provides an example of runtime tables. Runtime table 822 provides an example of a runtime table. Runtime table 822 can be used to store relationship information. Runtime table 822 has left-hand side 824 and right-hand side 826. Left-hand side 824 contains the left-hand side of a relationship (e.g., includes, removes, can't work with or excludes). The right-hand side of a relationship is stored in right-hand side 826.

Table 828 provides an example of runtime table 822. Table 828 can be an includes, excludes, or removes table, for example. Left-hand side 830 contains a bit vector (i.e., "01010 . . . 0"). Right-hand side 832 contains a bit vector (i.e., "10001 . . . 1"). Each bit in bit vectors 830 and 832 corresponds to an element in parts catalog 204 or a product. For example, if the product definition is for an automobile, each "1" bit in bit vector "01010 . . . 0" corresponds to a part in parts catalog 204 that is contained in an automobile's product definition. Zero bits indicate that the part is not included in the product definition. With reference to bit vectors, parts can be identified by the position of their corresponding bit in the bit vector. For example, the part (that corresponds with the "1" bit in the bit vector "0010" is part number three.

If table 830 is an instance of an includes table, when parts designated on left-hand side 830 (e.g., part numbers two and four) are selected, configuration system 212 includes the parts designated in right-hand side 832 (e.g., part numbers one and five). Similarly, if table 830 is excludes (or can't work with) table, parts one and five are excluded from the selectable parts when parts two and four are selected. Parts one and five are removed from the configuration when parts two and four are selected.

Like the includes, excludes, and removes tables, the requires choice table contains a left-hand side and a right-hand side. In one embodiment, both sides of the requires choice table contains a bit vector that corresponds to elements in parts catalog 204. Thus, for example, when selected, the part(s) identified on left-hand side 836 require a choice of a minimum and maximum number of parts from the parts contained on the right-hand side 838.

Table 822 can be altered to optimize the storage and use of information for a required choice table. Table 834 provides an example of an altered table 822. Preferably, however, right-hand side 838 contains a pointer to a group table (e.g., group table 840) that contains a bit vector (bit vector 842) that identifies the parts contained in the group. An advantage that the latter (altered table 822) format has over the former (unaltered table 822) is that it is only necessary to define the members of a group once. Each product definition that includes the group can point to the entry in the group table that contains the group definition.

One relationship table can be used for all products per relationship type. That is, a single includes table is used for includes relationships for all of the products. Alternatively, a table can be partitioned for each product. That is, the relationships associated with a product are segregated. For example, there would be a includes relationship table for each product. This has the advantage of only having to process only those relationships that exist for the chosen product. When a product is unselected or not selectable, there is no need to process the tables associated with that product. The latter alternative is preferable because it lessens processing requirements.

Configuration System

Using configuration system 212, a configuration user can configure a system given a product definition (e.g., product

definition 208) and part relationships associated with the product definition. Configuration system 212 accepts a configuration user's input and processes it to update the configuration based on the user's input or notify the user that the input is invalid given the current state of the configuration and the system definition. Configuration system 212 ensures that the current configuration state (determined by the product definition and order-independent user selection) is always valid.

User Selection

When user input is received by configuration system 212, it is processed based on the current configuration state, the product definition and the parts relationships. FIG. 9 provides a process flow for processing a user selection according to an embodiment of the invention.

At step 902, the user selects item n (e.g., part) preferably using a GUI screen. At step 904 (i.e., "n selectable?"), a determination is made whether the item is selectable. Thus, for example, configuration system 212 determines whether the nth bit in the selectable state information is set. If it is not, configuration system 212 determines that the item is not selectable by the user and raises an error at step 922. Processing then continues at step 914 to update the interface (if necessary) and return control of the GUI screen to the user. Processing of the current input ends at step 916.

If, however, it is determined at step 904 that the item selected by the user is selectable, processing continues at step 908 to set the nth bit in uVec 804. At step 910, the visible state information is updated as discussed above. At step 912, the relationships associated with the product definition are evaluated as discussed below. Processing continues at step 914 to update the interface (if necessary) and return control of the GUI screen to the user. Processing of the current input ends at step 916.

Relationship Evaluation

Relationships associated with items contained in the product definition are evaluated when user input is received. Configuration system 212 determines what relationships are active and inactive given the user input (at step 912, for example). A relationship is active when all the items on the left-hand side of the relationship are selected. A relationship is inactive until all of the parts on the left-hand side of the relationship are selected.

As previously discussed, relationship tables can be used to store the left-hand and right-hand sides of a relationship. That is, a relationship can be defined as a pair of bit vectors, the lhsVec (left-hand side) and the rhsVec (right-hand side). A relationship is made active if the following expression is true:

$$\text{SelectedVec} \ \& \ \text{lhsVec} = \text{lhsVec}$$

When a relationship is made active, updating the appropriate state vector (iVec, xVec, or rVec) becomes:

$$\text{stateVec} = \text{rhsVec}$$

An active relationship causes the configuration state to be updated. When an include relationship is activated, for example, parts identified on the right-hand side of the include relationship are added to the configuration, if they are not already included. The state is changed by modifying iVec 806. A change in iVec 806 can change other state information such as the selected and selectable state information. Parts that are included as a result of the activation of the includes relation can result in a ripple effect that causes other relationships to become activated. Thus, additional state changes can occur as a result of the include relationship's activation.

To further illustrate, when a requires choice relation becomes active, configuration system 212 includes and excludes items of the given group where applicable: if exactly max (maximum number of) items of the given group are selected, the engine excludes all other items in the group (added into xVec); otherwise, if less than min (minimum number of) items of the group are selected and only one valid selection path exists to make exactly min items selected, the engine includes (into iVec) the items that are needed (note that these must have been selectable items).

Configuration system 212 also ensures that no relationship that will put the configuration into an invalid state can become active (for example: if b is Selected then the relationship [a] can't work with [b] should not be active). If a relationship will make a configuration invalid, it is made notActivateable. For example, a relation should be made notActivateable, if:

- 1) an includes relationship has excluded (xVec) parts on the right-hand side;
- 2) a excludes (can't work with) relationship has Selected parts on the right-hand side;
- 3) a removes relation has user-selected (uVec) parts on the right side; or
- 4) a requires choice relation has a group for which between [min, max] parts can't be Selected.

In addition to the relationships created by a maintainer, there are some relationships that are dynamically created or deduced at runtime. If a relation becomes notActivateable and all but one item from the left-hand side is Selected, then the unSelected item is excluded by the engine. This ensures that the relation will not become active.

Configuration system 212 evaluates each relationship forward and backward. In a forward evaluation, the left-hand side of a relationship is examined to determine whether all of its parts are included in the configuration. If they are, the relationship is considered to be active. An active relationship causes the right-hand side of the relationship to then be evaluated. The evaluation is dependent on the type of relationship. An includes relationship causes the parts specified in the right-hand side to be included in the configuration specification. A removes relationship causes the right-hand side parts to be removed from the configuration. An excludes relationship cause the left-hand side parts to become notSelectable. A requires choice relationship recognizes that a choice has to be made by the user between the set of right-hand parts.

In a backward evaluation, configuration system 212 determines whether a relationship if activated would result in an invalid state. If so, configuration system 212 takes steps to ensure that the relationship cannot be activated. Instead of evaluating the left-hand side first as in the case of a forward evaluation, a backward evaluation evaluates the relationship by first examining the right-hand side. If a relationship fails the backward evaluation (i.e., its activation would result in an invalid state), configuration engine 212 takes steps to ensure that the relationship cannot become active. For example, if a relationship states that part A can't work with part B and part B is selected, configuration system 212 excludes part A. This ensures that part A cannot be selected and the excludes relationship cannot become active. Otherwise, a user can select part A which would result in an invalid configuration state.

The invention performs a relaxation over the set of relationships. A relaxation is an iterative process that is repeated until a state that has been defined no longer changes. That is, relationship evaluation is completed when none of iVec, xVec, or rVec are modified during an iteration.

The invention ensures that the relaxation process terminates by only adding bits to the state. That is, because iVec, xVec, and rVec all grow through the iteration, the relaxation process is guaranteed to terminate.

FIG. 10 provides an example of a relationship evaluation process flow according to an embodiment of the invention. At step 1002, rgVec is initialized to zero. During the relationship evaluation, rgVec is regenerated. At step 1004, copies of iVec, xVec, and rVec are made. The copies are used to determine whether relaxation is finished. At step 1006, the includes table associated with the product definition is evaluated in the forward direction. Similarly, forward evaluation is performed on the excluded and removed tables. The forward evaluation performs an "or" operation with the selectedVec and a specified configuration state vector (e.g., a forward evaluation of the includes table performs an "or" operation with the selectedVec and the iVec) when the left-hand side of the relationship is true. For example, when the items that correspond with the "1" bits in left-hand side 830 are all selected (i.e., in the selected vector), right-hand side 832 is "or'ed" with the iVec to add the items that correspond with the "1" bits in right-hand side 832 to the configuration where a forward evaluation of an includes relationship is being performed.

At step 1008, a backward evaluation is performed on the relationships contained in the includes, excludes, and removed tables. In a backward evaluation of an include relationship, the xVec is examined to determine whether any right-hand side items of an include relationship are excluded. If so, selection of all of the left-hand side items would result in an invalid configuration state. Therefore, the last remaining left-hand side item (if only one remains) is excluded to cause the relationship to never become active. Thus, the relationship is rendered notActivateable.

Backward evaluation of an excluded relationship examines the selectedVec. If a left-hand side item (an item that is supposed to be excluded when the relationship becomes active) is selected, selection of all of the left-hand side items would result in an invalid configuration state. To ensure against an invalid configuration state, a left-hand side item is excluded so that the relationship never becomes active (the relationship is rendered notActivateable).

In the preferred embodiment, configuration system 212 cannot delete, or remove, a user-selected item. The uVec is examined in a backward evaluation of a removes relationship. If a right-hand side item is a user selected item, the selection of all of the left-hand side items would result in an invalid configuration state. To ensure against an invalid configuration state, a left-hand side item is excluded so that the relationship becomes notActivateable.

At step 1010, backward and forward evaluation is performed on the requires choice table 834. In the backward evaluation, a determination is made whether a requires choice relationship should become notActivateable. For example, if it is impossible to satisfy a group, then a left-hand side item should be excluded to cause the relationship to become notActivateable.

In the forward evaluation, rgVec is set and items are included or excluded where appropriate. For example, if the right-hand side items are within range (within min and max values for the required choice relationship), then the remaining right-hand side items are excluded. If the only way to make a requires choice valid is to select the items remaining on the right-hand side item, configuration system 212 includes these items. In addition, a requires choice relationship is evaluated to determine whether the right-hand side items selected are within a valid range. If not, the corre-

sponding bit in the rgVec is set. If so, the corresponding bit in the rgVec is zeroed.

The selected, selectable, and notSelectable states are updated at step 1012. At step 1014 (i.e., "change in a vector?"), a determination is made whether any of iVec, xVec, or rVec were modified during the current iteration of the relaxation process. If not, a stable state has been achieved, and processing ends at step 1016. If so, additional iterations are needed to complete the relaxation process. Processing therefore continues at step 1002 to initiate a new iteration.

In evaluating relationships, it is not necessary to evaluate a relationship in either the forward or backward direction once it has become active. Also, when a relationship is notActivateable, there is no need to evaluate the relationship in the forward direction. When a relationship is notActivateable and has been excluded, there is no need to evaluate in the backward, or reverse, direction.

Unselection

FIG. 9 described processing where the user input was an item selection. Configuration system 212 also processes user input to unselect an item from the configuration. As previously discussed, an item can be unselected in any order. There is no requirement to unselect items the reverse order in which they were selected. FIG. 11 provides an example of an unselect item process flow according to an embodiment of the invention.

At step 1102 (i.e., "selection in uVec?"), a determination is made whether the item that is selected for deletion was selected for inclusion in the configuration by the user. If it is determined that the item is not a user-selected item, processing continues at step 1104 to raise an error. Processing continues at step 1114 to update the GUI (if necessary) and return control to the user. Processing of the current input ends at step 1116.

If it is determined at step 1102 that the item selected for unselection was selected by the user, processing continues at step 1106 to reset the bit in the uVec that corresponds to the unselected item (i.e., set the bit to zero). This unselects the item from the current configuration. At step 1108, the iVec, xVec, and rVec vectors are initialized (set to zero). Steps 1110, 1112 and 1114 correspond to steps 910, 912, and 914, respectively, of FIG. 9. Processing ends at step 1116.

Translation Between Representations

To ensure correct runtime behavior of the engine, a potentially incomplete user-defined set of relationships is converted into a set of complete runtime relationship tables. The external representation is translated into an internal representation. The invention maps the hierarchies contained in parts catalog 204 to relationships and maps relationships to the runtime tables, for example.

As discussed above, compiler 704 performs a mapping between product-level relationships and part-level relationships. The translation process further provides the ability to translate a multi-level hierarchy to a single-level hierarchy. That is, the translation process can be used to flatten out the intuitive hierarchy created by a maintenance user using the user interface of FIG. 6, for example. FIG. 12 provides a flow for translation processing according to an embodiment of the invention.

At step 1202, a set of relationships is created based on the hierarchies. Thus, for example, group 312 is translated into an requires choice relationship. That is, group 312 requires a choice between parts 306, 308, and 310. Relationships can be defined in terms of groups. An identifier (item ID) is used to identify the group. When a group appears on the left-hand side of a relationship, the group is assigned an item ID and

the relationship is defined in terms of the new item ID. This process, referred to as back-linking, is performed at step 1204.

At step 1206, an include chain evaluation is performed. In an include chain evaluation, all include relationship chains are fully evaluated so that relationship contains all of the parts that are included in part regardless of whether the included relationship is direct (e.g., mentioned in the includes relationship) or indirect (mentioned in another includes relationship that contains an included part on the right-hand side).

At step 1208, subgroup excludes are evaluated. In this process, active requires choice relationships are evaluated where one relationship's right-hand side is a subset of another relationship's right-hand side. In this case, if both relationships are active, the right-hand side parts that are not within the subset are excluded. Relationship factorization is performed at step 1210. Relationship factorizing can create new relationships that ensure proper runtime behavior when more than one item is selected during some iteration.

Removing redundancy in relations may cause some relations to become unnecessary. A relation becomes unnecessary when it contains no items on its right side. Redundant relationships are removed at step 1212. Processing ends at step 1214.

Back-Linking

Relationships can be defined in terms of groups. When a group appears on the left-hand side of a relationship, the group is assigned an item ID and behaves as a normal item (e.g., part) and the relationship is defined in terms of this new item ID. A relationship is defined for each item in the group:

item includes item ID (of the group)

It is only necessary to create this relationship once per group, regardless of how many times that group appears on the left side of a relationship. Back-Linking can be used to generate relationships in nested groups. FIGS. 13A-13B provide an illustration of groups and nested groups according to an embodiment of the invention. Group 1302 contains two items (groups 1304 and 1306) that are nested within group 1302. Notation is provided for a group to specify the minimum and maximum bounds for a group. For example, the notation "(0,1)" indicates that between zero and one items in group 1302 are to be chosen.

The items contained in group 1302 are also groups that contain multiple items. Groups 1304 and 1306 are nested within group 1302. Two or three of parts 1308-1312 can be chosen from Group 1304. Zero to three of parts 1314-1318 can be chosen from Group 1306.

The invention creates a metapart for these groups (groups 1302-1306). Relationships are then created between the groups and their members such that items are back-linked to their parents. FIG. 13B provides an illustration of metaparts for the groups and nested group of FIG. 13A according to an embodiment of the invention. Groups 1302'-1306' are metaparts for groups 1302-1306, respectively. Relationships can be generated using these metaparts as follows:

Group 1302' requires choice [Group 1304', Group 1306'] 0,1

Group 1304' requires choice [part 1308, part 1310, part 1312] 2,3

Group 1306' requires choice [part 1314, part 1316, part 1318] 0,3

The metapart is used internally; it is unnecessary for a user to be aware of them. Using back-linking, relationships can be generated such that an item (e.g., part 1308) links back to its predecessor(s) (e.g., nested group, group 1302).

The following new relationships can be created for the examples of FIGS. 13A-13B:

Group 1304' includes Group 1302'

Group 1306' includes Group 1302'

Part 1308 includes Group 1304'

Part 1310 includes Group 1304'

Part 1312 includes Group 1304'

Part 1314 includes Group 1306'

Part 1316 includes Group 1306'

Part 1318 includes Group 1306'

When a user selects part 1308, part 1308 includes group 1304'. Group 1304' is added to the configuration. Further, group 1304' includes group 1302' thereby including group 1302'. When group 1302' is included in the configuration, it activates the relationship that requires a choice between groups 1304' and 1306'. Since group 1304' is already selected, group 1306' can be excluded. When group 1306' is excluded, parts 1314-1318 are excluded when the newly generated relationships are evaluated in the reverse (backward) direction. Further, the bounds specified by group 1304 are also enforced using these relationships.

As discussed above, a requires choice relationship can be defined at the product level. A product can require a choice between items in a group. The product-level requires choice relationship is treated similarly to that of a requires choice part-level relationship.

Include Chains

In an include chain evaluation, all include relationship chains are fully evaluated so that a relationship can be generated that contains all of the parts that are included regardless of whether the relationship is direct (e.g., the included parts mentioned in the includes relationship) or indirect (a part is mentioned in another includes relationship that contain an included part on the right-hand side). For example, the following relationships exists in the external representation:

a includes b and c

b includes d

d and c includes e

An include chain exists that starts with a. The first relationship indicates that a includes b and c. Based on the second and third relationships, b and c include other parts. Based on all of the relationships, a includes b, c, d, and e. Include chain evaluation expands these relationships into the following relationships:

a includes b and c and d and e

b includes d

d and c includes e

It is not necessary to generate additional relationships to perform includes chain evaluation. Preferably, includes chain evaluation can modify an existing relationship to include both direct and indirect includes relationships. Each includes relationship can be evaluated against all other includes relationships to determine whether its right-hand side should be modified to include both direct and indirect includes relationships. Once all of the includes relationships have been evaluated, a new right-hand side value can be generated based on the evaluation. FIGS. 14A-14B provide an includes chain process flow according to an embodiment of the invention.

At step 1402 (i.e., "all include rels processed?"), a determination is made whether all of the includes relationships have been processed. If other includes relationships remain to be processed, processing continues at step 1408 to get the next includes relationship, r. At step 1410, a temporary

vector, temp, is created that is the result of an "or" operation between the left-hand and right-hand sides of r. Another temporary vector, tempsave, is created that is set to the value of temp at step 1412. At step 1414 (i.e., "all rels. compared?"), a determination is made whether all of the other includes relationships have been evaluated against r. If not, processing continues at step 1418 to get another relationship, r₂. At step 1420 (i.e., "temp hits r₂?"), a determination is made whether any of the items contained in temp are also contained in r₂. If not, processing continues at step 1414 to process any remaining relationships against r. If so, processing continues at step 1422 to perform a "or" operation between temp and the right-hand side of r₂ to add the right-hand side of r₂ to the temp vector.

If it is determined, at step 1414, that all of the relationships r₂ have been evaluated for r, processing continues at step 1416. At step 1416 (i.e., "temp=tempsave?"), a determination is made whether any items were added to the temp vector. If not, there are no indirect includes and processing continues at step 1424 to reset r₂ and processing continues at step 1414 to process any remaining relationships against this relationship. If so, processing continues at step 1404 to modify the right-hand side of r to include the items contained in the temp vector minus the items contained in the left-hand side of r. Processing continues at step 1402 to process any remaining relationships.

If it is determined at step 1402 (i.e., all include rels. processed?) that all of the includes relationships have been evaluated for include chains, processing ends at step 1404.

Subgroup Excludes

The subgroup excludes evaluation is performed on requires choice relationships. Assume a requires choice relationship points to a group (A) that is a complete subgroup of a group (B) of another requires choice relationship. If B has a max of 1 and both relationships are made active, the items in B that are not in A, Q, cannot be chosen. If a pairwise comparison of requires choice relationships yields such cases, a relationship such as the following is generated:

[union of left sides of both relationships] can't work with Q

For example, assume that the following relationships have been defined:

a b requires choice [x, y, z] (0,1)

d requires choice [x, y] (0,2)

The second relationship contains a group that is a subgroup of the first relationship. When both relationships are active, the non-subset is excluded. This avoids the case where a user selects a, b, and d and the relationship evaluations do not exclude the items not included in both groups (the non-subset). The invention combines these relationships and creates an excludes relationship. The excludes relationship for the above example is as follows:

a b d excludes z

FIGS. 15A-15B provides an example of a subgroup excludes process flow according to an embodiment of the invention. At step 1502 (i.e., "all rels. processed?"), a determination is made whether all of the requires choice relationships have been processed. If so, processing ends at step 1504. If not, processing continues at step 1506 to get the next relationship, r. At step 1507 (i.e., "r.rhs.max=1?"), a determination is made whether the maximum value for the relationship is equal to one. If not, processing continues at step 1502 to process any remaining relationships. If so, processing continues at step 1508. At step 1508 (i.e., "all rels. compared?"), a determination is made whether all of the other requires choice relationships have been processed against r. If so, processing continues at step 1502 to process another requires choice relationship.

If not, processing continues at step 1510 to get another relationship, r_2 . At step 1512 (i.e., " $r=r_2$?"), a determination is made whether the two relationships are equal. If so, processing continues at step 1508 to process any remaining requires choice relationships against the current r . If r and r_2 are not equal, processing continues at 1514. At step 1514 (i.e., " $r_2.grp$ subset of $r.grp$?"), a determination is made whether the right-hand side of r_2 is a subset of the right-hand side of r . If not, processing continues at step 1508 to process any remaining requires choice relationships against the current r . If so, processing continues at step 1516 to create an exclude relationship where the left-hand side is the union of the left-hand side of r and r_2 and the right-hand side is the non-subset of the right-hand side of the two relationships. Processing continues at step 1508 to process any remaining requires choice relationships against the current r .

Relationship Factorization

The invention assumes that one item on the left side of a relationship is preferably selected during any iteration. However, the includes and requires choice relationships may cause additional items to become selected (e.g., selected by the system). Factorizing of relationships can create new relationships to ensure proper runtime behavior when more than one item is selected during some iteration.

Factorization causes a relationship to be created where a relationship has items on the left-hand side that also appear on the right-hand side of an includes relationship. The new relationship represents the common factorization of both relationships. For example, where the following two relationships exist:

[a] includes [b and c]

[b and c and d and e] can't work with [x and y]

the following relationship is created:

[a and d and e] can't work with [x and y]

The invention uses a technique in factorization that avoids combinatorial explosion. Relationships are created during factorization when they add value. The invention identifies value where the new relationship:

1. replaces more than one item from the left-hand side;
2. has at least two different includes relationships replace at least one unique item from the left-hand side of the relationship being factorized; and
3. the two includes relationships have at least one common item in their left-hand side.

FIGS. 16A-16C provide an example of a relationship factorization process flow according to an embodiment of the invention. At step 1602, each relationship is marked as "new". At step 1604, a state variable, added, is initialized to "false". At step 1606 (i.e., "all includes rels. processed?"), a determination is made whether all of the includes relationships have been processed. If so, processing continues at step 1608 (i.e., "added='true'?") to determine whether any new relationships were added. If not, processing ends at step 1610. If there are new relationships, processing continues at step 1609 to reset the includes relationships and processing continues at step 1604 to initialize added to "false".

If it is determined at step 1606 that all of the includes relationships have not been processed, processing continues at step 1612 to get the next includes relationship, i . At step 1613, the relationships are reset. At step 1614 (i.e., "all 'new' rels. processed?"), a determination is made whether all of the relationships have been marked as "new" have been processed against i . If so, processing continues at step 1606 to process any remaining includes relationships. If not, processing continues at step 1616.

At step 1616, the next relationship, r , is marked as "old". At step 1620 (i.e., " $!(r.lhs \&\& i.rhs)$?"), a determination is

made whether the right-hand side of the i relationship and the left-hand side of the r relationship have any bits in common. If they do not have bits in common, processing continues at step 1614 to process any remaining relationships against the current i . If they do have bits in common, processing continues at step 1622 to generate new.lhs from the includes relationship and the left-hand side of r . Step 1622 takes out items in the right-hand side of i from left-hand side of r and adds in items that are in the left-hand side of i . At step 1624 (i.e., "new.lhs has value?") a determination is made whether new.lhs has value. This determination can use the guidelines as specified above.

If new.lhs does not have value, processing continues at step 1614 to processing any remaining relationships against i . If it does have value, processing continues at step 1626 to add a new relationship (of the type r) that uses new.lhs as the left-hand side and the right-hand side of r . The new relationship is marked as "new" at step 1628. The value of added is set to "true" at step 1630. Processing continues at step 1614 to process any remaining relationships.

Relationship Compression

Removing redundancy in relations may cause some relations to become unnecessary. A relation becomes unnecessary when it contains no items on its right side. If a relationship's left-hand side is a proper subset of the left-hand side of another relationship, then the first relationship is guaranteed to be active whenever the second relationship is active. If both relationships are also the same type of relationship, the common right-hand side elements can be removed from the second relationship. This can be repeated to remove all unnecessary relationships. The following relationships provide an example:

[a] removes [c]

[a and b] removes [c and d]

The left-hand side of the first relationship is a proper subset of the left-hand side of the second relationship (e.g., all elements "a" in the left-hand side of the first relationship are contained in the left-hand side of the second relationship). The relationships are the same type. Therefore, the common right-hand side elements can be removed from the right-hand side of the second relationship. The following is the result after modifying the relationship:

[a] removes [c]

[a and b] removes [d]

FIGS. 17A-17B provide an illustration of a relationship compression process flow according to an embodiment of the invention. At step 1702 (i.e., "all relationships in table processed?"), a determination is made whether all of the relationships in the table have been processed. If not, processing continues at step 1704 to get the next relationship, r . At step 1706 (i.e., " r processed against all other rels. in table?"), a determination is made whether the other relationships in the table have been processed against r . If so, processing continues at step 1702 to process any remaining relationships in the table as r .

If it is determined at step 1706 that all of the other relationships have not been processed against r , processing continues at step 1708. Another relationship is retrieved as r_2 at step 1708. At step 1710 (i.e., " $r=r_2$?"), a determination is made whether the relationships are the same. If not, processing continues at step 1706 to process any remaining relationships against r . If so, processing continues at step 1712. At step 1712 (i.e., " $r.lhs$ subset of $r_2.lhs$?"), a determination is made whether the left-hand side of r is a subset of the left-hand side of r_2 . If the subset condition is true, processing continues at step 1714 to remove the common items out of the right-hand side of r (i.e., perform the

operation $r_2.rhs = r_2.rhs$ and not $r.rhs$). Processing continues at step 1706 to process any remaining relationships against r .

If it is determined at step 1702 that all of the relationships have been processed as r , processing continues at step 1716. At step 1716 (i.e., "all relationships in table processed?") a determination is made whether all of the relationships have been processed. If so, processing ends at step 1718. If not, processing continues at step 1720 to get another relationship as r . At step 1722 (i.e., " $r.rhs = \text{zeroes?}$ "), a determination is made whether the right-hand side of r has been set to zero has a result of compression. If not, processing continues at step 1716 to process any remaining relationships. If so, processing continues at step 1724 to delete r from the table. Processing continues at step 1716 to process any remaining relationships.

Global Relationships

Global relationships can be defined that are processed for all products. Thus, it is unnecessary to redefining a relationship for all of the products. The relationship can be defined as a global relationship that is applied for every product. If, however, partitioned relationship tables are used for each product, it is necessary to evaluate the relationships in a product's partitioned relationship tables as well as the global relationships. For example, the process flows described herein that operate on relationship tables can be performed once for the product's partitioned relationship tables and once for the global relationships.

Thus, a method and apparatus for maintaining and configuring systems has been provided.

We claim:

1. A method of configuring a system comprising the steps of:

generating a definition for said system, said definition containing one or more elements and being conveyed graphically using a set of product relationships, said product relationships identifying classifications for said one or more elements, said product relationships comprising an includes classification;

generating a set of part relationships between said one or more elements, said set of part relationships being conveyed graphically; and

configuring said system using said definition and said set of product relationships and said part relationships.

2. The method of claim 1 wherein said step of configuring further comprises the steps of:

receiving input from a configuration user;

validating said input based on said definition, said set of relationships, and a current configuration state; and

identifying a set of valid configuration options using said definition, said set of relationships and said current configuration state.

3. The method of claim 2 wherein said step of identifying further comprises the steps of:

determining whether a relationship in said set of relationships is active;

modifying said configuration to satisfy said active relationship when it is determined that a modification is needed;

making said relationship notActivateable when it is determined that said relationship is not active and the activation of said relationship would render the configuration invalid.

4. The method of claim 3 wherein said step of determining whether a relationship is active further comprises the step of determining whether the elements specified in the left-hand side of said relationship are present in said configuration.

5. The method of claim 4 wherein said step of determining whether the elements specified in the left-hand side of said relationship are present in said configuration further comprises the step of performing a bit-level comparison of the bits corresponding to the elements in said left-hand side of the relationship with a "selected" bit vector.

6. The method of claim 3 wherein said relationship is an includes relationship, said step of modifying further comprises the step of including in said configuration elements identified in a right-hand side of said relationship.

7. The method of claim 3 wherein said relationship is an excludes relationship, said step of modifying further comprises the step of marking said elements identified in the right-hand side of said relationship as excluded and notSelectable.

8. The method of claim 3 wherein said relationship is a removes relationship, said step of modifying further comprises the step of removing said elements identified in the right-hand side of said relationship from said configuration.

9. The method of claim 3 wherein said relationship is a requires choice relationship, said step of modifying further comprises the steps of:

determining whether a plurality of elements identified in a right-hand side of said relationship are present in said configuration such that said relationship is satisfied;

modifying said configuration to satisfy said requires choice relationship when it is determined that only one path exists to satisfy said relationship and said relationship is not already satisfied.

10. The method of claim 3 wherein said step of making said relationship notActivateable further comprises the step of marking one or more elements in said left-hand side of said relationship as notSelectable.

11. The method of claim 2 wherein said definition, each of said set of relationships, and said current configuration state are expressed as bit vectors having bits that correspond to elements.

12. The method of claim 2 wherein said step of validating further comprises the step of determining whether the relationships other than those marked notActivateable in said set of relationships can be satisfied.

13. The method of claim 2 wherein said step of generating a definition selects said one or more elements for inclusion in said definition, said input is made without regard for the order in which said one or more elements are selected for said definition.

14. The method of claim 2 wherein said configuration state identifies a set of selected elements and a set of selectable elements for said system, said input capable of selecting any member of said set of selectable elements and being made despite the order in which members of said set of selected elements were selected.

15. The method of claim 1 wherein a parts catalog is used to store information associated with said one or more elements, said information including an identifier and description.

16. A method for defining a system that can be configured comprising the steps of:

identifying an element of said system;

specifying said element as an included element when said element is automatically included in the configuration;

specifying said element as an optional element when said element is not a necessary component of said system;

specifying said element as a required choice when said element is a group that contains one or more members from which to choose.

23

17. The method of claim 16 wherein said element is a member of a first element set said first element set containing one or more elements, said method further comprising the steps of:

- identifying a second element set of said system, said second element set containing one or more elements; and
- specifying a relationship between said first element set and said second element set.

18. The method of claim 17 wherein said step of specifying further comprises the steps of:

- specifying that said second element set is to be included in said system when said first element set is present in said system;
- specifying that said second element set is to be excluded from said system when said first element set is present in said system; and
- specifying that said second element set is to be removed from said system when said first element set is present in said system.

19. The method of claim 16 further comprising the steps of:

- specifying a group of elements;
- specifying a relationship between said first element set and said group of elements such that one or more elements in said group of elements must be chosen when said first element set is present in said system.

20. An article of manufacture comprising:

- a computer usable medium having computer readable program code embodied therein for configuring a system comprising:

- computer readable program code configured to cause a computer to generate a definition for said system, said definition containing one or more elements and being conveyed graphically using a set of product relationships, said product relationships identifying classifications for said one or more elements, said product relationships comprising an includes classification;

- computer readable program code configured to cause a computer to generate a set of part relationships between said one or more elements, said set of part relationships being conveyed graphically; and

- computer readable program code configured to cause a computer to configure said system using said definition and said set of product relationships and said part relationships.

21. The article of manufacture of claim 20 wherein said computer readable program code configured to cause a computer to configure further comprises:

- computer readable program code configured to cause a computer to receive input from a configuration user;
- computer readable program code configured to cause a computer to validate said input based on said definition, said set of relationships, and a current configuration state; and

- computer readable program code configured to cause a computer to identify a set of valid configuration options using said definition, said set of relationships and a current configuration state.

22. The article of manufacture of claim 21 wherein said computer readable program code configured to cause a computer to identify further comprises:

- computer readable program code configured to cause a computer to determine whether a relationship in said set of relationships is active;

24

computer readable program code configured to cause a computer to modify said configuration to satisfy said active relationship when it is determined that a modification is needed;

computer readable program code configured to cause a computer to make said relationship notActiveable when it is determined that said relationship is not active and the activation of said relationship would render the configuration invalid.

23. The article of manufacture of claim 22 wherein said computer readable program code configured to cause a computer to determine whether a relationship is active further comprises computer readable program code configured to cause a computer to determine whether the elements specified in the left-hand side of said relationship are present in said configuration.

24. The article of manufacture of claim 23 wherein said computer readable program code configured to cause a computer to determine whether the elements specified in the left-hand side of said relationship are present in said configuration further comprises computer readable program code configured to cause a computer to perform a bit-level comparison of the bits corresponding to the elements in said left-hand side of the relationship with a "selected" bit-vector.

25. The article of manufacture of claim 22 wherein said relationship is an includes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to include in said configuration elements identified in a right-hand side of said relationship.

26. The article of manufacture of claim 22 wherein said relationship is an excludes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to mark said elements identified in the right-hand side of said relationship as excluded and notSelectable.

27. The article of manufacture of claim 22 wherein said relationship is a removes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to remove said elements identified in the right-hand side of said relationship from said configuration.

28. The article of manufacture of claim 22 wherein said relationship is a requires choice relationship, said computer readable program code configured to cause a computer to modify further comprises:

- computer readable program code configured to cause a computer to determine whether a plurality of elements identified in a right-hand side of said relationship are present in said configuration such that said relationship is satisfied;

- computer readable program code configured to cause a computer to modify said configuration to satisfy said requires choice relationship when it is determined that only one path exists to satisfy said relationship and said relationship is not already satisfied.

29. The article of manufacture of claim 22 wherein said computer readable program code configured to cause a computer to make said relationship notActiveable further comprises computer readable program code configured to cause a computer to mark one or more elements in said left-hand side of said relationship as notSelectable.

30. The article of manufacture of claim 21 wherein said definition, each of said set of relationships, and said current configuration state are expressed as bit vectors having bits that correspond to elements.

31. The article of manufacture of claim 21 wherein said code configured to cause a computer to validate further comprises program code configured to cause a computer to determine whether the relationships other than those marked notActivateable in said set of relationships can be satisfied.

32. The article of manufacture of claim 21 wherein said code configured to cause a computer to generate a definition selects said one or more elements for inclusion in said definition, said input is made without regard for the order in which said one or more elements are selected for said definition.

33. The article of manufacture of claim 21 wherein said configuration state identifies a set of selected elements and a set of selectable elements for said system, said input capable of selecting any member of said set of selectable elements and being made despite the order in which members of said set of selected elements were selected.

34. The article of manufacture of claim 17 wherein a parts catalog is used to store information associated with said one or more elements, said information including an identifier and description.

35. An article of manufacture comprising
a computer usable medium having computer readable program code embodied therein for defining a system that can be configured comprising:
computer readable program code configured to cause a computer to identify an element of said system;
computer readable program code configured to cause a computer to specify said element as an included element when said element is automatically included in the configuration;
computer readable program code configured to cause a computer to specify said element as an optional element when said element is not a necessary component of said system;
computer readable program code configured to cause a computer to specify said element as a required choice when said element is a group that contains one or more members from which to choose.

36. The article of manufacture of claim 35 wherein said element is a member of a first element set said first element set containing one or more elements, said article of manufacture further comprising:

computer readable program code configured to cause a computer to identify a second element set of said system, said second element set containing one or more elements; and

computer readable program code configured to cause a computer to specify a relationship between said first element set and said second element set.

37. The article of manufacture of claim 36 wherein said computer readable program code configured to cause a computer to specify further comprises:

computer readable program code configured to cause a computer to specify that said second element set is to be included in said system when said first element set is present in said system;

computer readable program code configured to cause a computer to specify that said second element set is to be excluded from said system when said first element set is present in said system; and

computer readable program code configured to cause a computer to specify that said second element set is to be removed from said system when said first element set is present in said system.

38. The article of manufacture of claim 35 further comprising:

computer readable program code configured to cause a computer to specify a group of elements;

computer readable program code configured to cause a computer to specify a relationship between said first element set and said group of elements such that one or more elements in said group of elements must be chosen when said first element set is present in said system.

39. A method of configuring a system comprising the steps of:

translating a system definition from an external representation comprising a set of intuitive relationships between components of said system definition into an internal representation comprising a set of internal relationships;

validating configuration input using said internal representation; and

modifying a configuration state based on said user input.

40. The method of claim 39 wherein said step of translating further comprises the step of:

removing a plurality of hierarchical levels from said external representation such that said internal representation is a single level.

41. The method of claim 40 wherein said step of translating further comprises the step of:

creating a set of relationships based on said hierarchical levels, each one in said set of relationships having a first set of components and a second set of components and a relationship operator.

42. The method of claim 41 wherein said relationship operator is chosen from the operators include, exclude, remove, and requires choice.

43. The method of claim 39 wherein each of said internal relationships has a left-hand side, a relationship operator and a right-hand side, said step of translating further comprises the steps of:

determining whether a group of components exists on said left-hand side of one of said internal relationships;

performing the following steps when a group of components exist on said left-hand side:

creating a new component for said group; and

creating a new relationship for each component of said group of components such that each component will include said new component.

44. The method of claim 39 wherein a type of said internal relationships is an include relationship, said include relationship identifies a second set of components that are included in said system when all of a first set of components exist in said system, said step of translating further comprises the steps of:

determining whether a subset of the union of said first and second sets of components in a first includes relationship comprises a first set of components in a second includes relationship; and

generating a combined includes relationship, said combined relationship comprises said first set of components of said first includes relationship and said second set of components from said first and second includes relationships when said subset of the union of said first and second sets of components in said first includes relationship comprises a first set of components in a second includes relationship.

45. The method of claim 39 wherein a type of said internal relationships is a requires choice relationship, said requires choice relationship identifies a second set of components

from which a number of components must be chosen when a first set of components exist in said system definition, said step of translating further comprises the steps of:

determining whether a second set of components of a second requires choice relationship is a proper subset of a second set of components of a first requires choice relationship;

performing the following steps when said condition exists:

creating a new relationship such that said new relationship excludes a second set of components of said new relationship when a first set of components of said new relationship is selected, said new relationship's first set of components is the union of the first set of components of said first and second requires choice relationships and said new relationship's second set of components is the difference of the second set of components of said first and second requires choice relationship.

46. The method of claim 39 wherein a type of said internal relationships is an includes relationship, said includes relationship identifies a second set of components that are included in said system when all of a first set of components exist in said system, said step of translating further comprises the steps of:

determining whether the second set of components of an includes relationship are contained in a first set of components of another internal relationship;

replacing said second set of components of said includes relationship in said another internal relationship with said first set of components of said includes relationship when said second set of components of an includes relationship are contained in a first set of components of another internal relationship.

47. The method of claim 39 said internal representation comprises multiple types of relationships, each type of relationship having a first and second set of components and an operation, said operation is performed on said second set of components when said first set of components exist in said system.

48. The method of claim 47 wherein said step of translating further comprises the steps of:

determining whether a first and second relationship are the same type of relationship;

performing the following steps when they are the same type:

determining whether the first relationship's first set of components is a proper subset of the second relationship's first set of components;

removing the first relationship's second set of components that exist from the second relationship's second set of components when said proper subset exists;

removing said second relationship when said second relationship's second set of components is an empty set.

49. The method of claim 47 wherein said step of validating further comprises the steps of:

determining whether a component selected by said configuration input is selectable; and

invalidating said configuration input when said component is not selectable.

50. The method of claim 49 wherein said one of said internal relationships is an includes relationship, said includes relationship causing a second set of components to be included in said system when all members of a first set of

components exist in said system, said second step of determining further comprises the step of:

determining whether any member of the second set of components cannot be included in said system.

51. The method of claim 49 wherein said one of said internal relationships is an excludes relationship, said excludes relationship causing a second set of components to be excluded from said system when all members of a first set of components exist in said system, said second step of determining further comprises the step of:

determining whether any member of the second set of components already exists in said system.

52. The method of claim 49 wherein said one of said internal relationships is an removes relationship, said removes relationship causing a second set of components to be removed from said system when all members of a first set of components exist in said system, said second step of determining further comprises the step of:

determining whether any member of the second set of components has been selected for inclusion in said system by a configuration user.

53. The method of claim 49 wherein said one of said internal relationships is a requires choice relationship, said requires choice relationship requiring a selection of a number of members of a second set of components when all members of a first set of components exist in said system, said second step of determining further comprises the step of:

determining whether said number of members of said second set of components cannot be selected for addition to said system.

54. The method of claim 47 wherein said step of modifying further comprises the steps of:

determining whether the performance of said operation of one of said internal relationships would result in an invalid configuration;

causing said one of said internal relationships to become inactivateable.

55. The method of claim 54 wherein said step of causing further comprises the step of excluding the last remaining member of said first set of components such that said last remaining member cannot be added to said system, said last remaining member being the only member of said first set of components that has not already been selected.

56. The method of claim 39 wherein said step of translating further comprises the step of:

mapping said intuitive relationships of said external representation into a set of relationships in said internal representation.

57. The method of claim 39 wherein said internal representation is stored as a set of bit vectors wherein each bit vector includes a bit that corresponds to one of said components of said system definition, the impact of changing the number of said components of said system definition thereby being minimal.

58. The method of claim 57 wherein said step of validating comprising a plurality of bit-level operations.

59. The method of claim 39 wherein said step of modifying further comprises the step of updating said configuration state to represent which of said components of said system definition are selectable, selected, included, and excluded.

60. A method of configuring a system comprising the steps of:

generating a definition for said system, said definition containing one or more elements and being conveyed graphically using a set of product relationships;

generating a set of part relationships between said one or more elements, said set of part relationships being conveyed graphically using a set of part relationships; and

receiving input from a configuration user;

validating said input based on said definition, said set of product relationships, said set of part relationships, and a current configuration state; and

identifying a set of valid configuration options using said definition, said set of product relationships, said set of part relationships and said current configuration state.

61. The method of claim 60 wherein said step of identifying further comprises the steps of:

determining whether a relationship in said set of relationships is active;

modifying said configuration to satisfy said active relationship when it is determined that a modification is needed;

making said relationship notActivateable when it is determined that said relationship is not active and the activation of said relationship would render the configuration invalid.

62. The method of claim 61 wherein said step of determining whether a relationship is active further comprises the step of determining whether the elements specified in the left-hand side of said relationship are present in said configuration.

63. The method of claim 62 wherein said step of determining whether the elements specified in the left-hand side of said relationship are present in said configuration further comprises the step of performing a bit-level comparison of the bits corresponding to the elements in said left-hand side of the relationship with a "selected" bit vector.

64. The method of claim 61 wherein said relationship is an includes relationship, said step of modifying further comprises the step of including in said configuration elements identified in a right-hand side of said relationship.

65. The method of claim 61 wherein said relationship is an excludes relationship, said step of modifying further comprises the step of marking said elements identified in the right-hand side of said relationship as excluded and notSelectable.

66. The method of claim 61 wherein said relationship is a removes relationship, said step of modifying further comprises the step of removing said elements identified in the right-hand side of said relationship from said configuration.

67. The method of claim 61 wherein said relationship is a requires choice relationship, said step of modifying further comprises the steps of:

determining whether a plurality of elements identified in a right-hand side of said relationship are present in said configuration such that said relationship is satisfied;

modifying said configuration to satisfy said requires choice relationship when it is determined that only one path exists to satisfy said relationship and said relationship is not already satisfied.

68. The method of claim 61 wherein said step of making said relationship notActivateable further comprises the step of marking one or more elements in said left-hand side of said relationship as notSelectable.

69. The method of claim 60 wherein said definition, each of said set of relationships, and said current configuration state are expressed as bit vectors having bits that correspond to elements.

70. The method of claim 60 wherein said step of validating further comprises the step of determining whether the

relationships other than those marked notActivateable in said set of relationships can be satisfied.

71. The method of claim 60 wherein said step of generating a definition selects said one or more elements for inclusion in said definition, said input is made without regard for the order in which said one or more elements are selected for said definition.

72. The method of claim 60 wherein said configuration state identifies a set of selected elements and a set of selectable elements for said system, said input capable of selecting any member of said set of selectable elements and being made despite the order in which members of said set of selected elements were selected.

73. An article of manufacture comprising:

a computer usable medium having computer readable program code embodied therein for configuring a system comprising:

computer readable program code configured to cause a computer to generate a definition for said system, said definition containing one or more elements and being conveyed graphically using a set of product relationships;

computer readable program code configured to cause a computer to generate a set of part relationships between said one or more elements, said set of part relationships being conveyed graphically using a set of part relationships; and

computer readable program code configured to cause a computer to receive input from a configuration user; computer readable program code configured to cause a computer to validate said input based on said definition, said set of product relationships, said set of part relationships, and a current configuration state; and

computer readable program code configured to cause a computer to identify a set of valid configuration options using said definition, said set of product relationships, said set of part relationships and said current configuration state.

74. The article of manufacture of claim 73 wherein said computer readable program code configured to cause a computer to identify further comprises:

computer readable program code configured to cause a computer to determine whether a relationship in said set of relationships is active;

computer readable program code configured to cause a computer to modify said configuration to satisfy said active relationship when it is determined that a modification is needed;

computer readable program code configured to cause a computer to make said relationship notActivateable when it is determined that said relationship is not active and the activation of said relationship would render the configuration invalid.

75. The article of manufacture of claim 74 wherein said computer readable program code configured to cause a computer to determine whether a relationship is active further comprises computer readable program code configured to cause a computer to determine whether the elements specified in the left-hand side of said relationship are present in said configuration.

76. The article of manufacture of claim 75 wherein said computer readable program code configured to cause a computer to determine whether the elements specified in the left-hand side of said relationship are present in said configuration further comprises computer readable program code configured to cause a computer to perform a bit-level

31

comparison of the bits corresponding to the elements in said left-hand side of the relationship with a "selected" bit-vector.

77. The article of manufacture of claim 74 wherein said relationship is an includes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to include in said configuration elements identified in a right-hand side of said relationship.

78. The article of manufacture of claim 74 wherein said relationship is an excludes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to mark said elements identified in the right-hand side of said relationship as excluded and notSelectable.

79. The article of manufacture of claim 74 wherein said relationship is a removes relationship, said computer readable program code configured to cause a computer to modify further comprises computer readable program code configured to cause a computer to remove said elements identified in the right-hand side of said relationship from said configuration.

80. The article of manufacture of claim 74 wherein said relationship is a requires choice relationship, said computer readable program code configured to cause a computer to modify further comprises:

computer readable program code configured to cause a computer to determine whether a plurality of elements identified in a right-hand side of said relationship are present in said configuration such that said relationship is satisfied;

computer readable program code configured to cause a computer to modify said configuration to satisfy said

32

requires choice relationship when it is determined that only one path exists to satisfy said relationship and said relationship is not already satisfied.

81. The article of manufacture of claim 74 wherein said computer readable program code configured to cause a computer to make said relationship notActivateable further comprises computer readable program code configured to cause a computer to mark one or more elements in said left-hand side of said relationship as notSelectable.

82. The article of manufacture of claim 73 wherein said definition, each of said set of relationships, and said current configuration state are expressed as bit vectors having bits that correspond to elements.

83. The article of manufacture of claim 73 wherein said code configured to cause a computer to validate further comprises program code configured to cause a computer to determine whether the relationships other than those marked notActivateable in said set of relationships can be satisfied.

84. The article of manufacture of claim 73 wherein said code configured to cause a computer to generate a definition selects said one or more elements for inclusion in said definition, said input is made without regard for the order in which said one or more elements are selected for said definition.

85. The article of manufacture of claim 73 wherein said configuration state identifies a set of selected elements and a set of selectable elements for said system, said input capable of selecting any member of said set of selectable elements and being made despite the order in which members of said set of selected elements were selected.

* * * * *



US005870719A

United States Patent [19]

Maritzen et al.

[11] Patent Number: **5,870,719**[45] Date of Patent: ***Feb. 9, 1999**

[54] **PLATFORM-INDEPENDENT, USAGE-INDEPENDENT, AND ACCESS-INDEPENDENT DISTRIBUTED QUOTE CONFIGURATON SYSTEM**

[75] Inventors: **L.M. Maritzen; Rolando D. Dimaandal**, both of Milpitas; **Julla Giannella**, Saratoga; **Raul Arregui**, Pacifica; **Marc Moss**, Fremont, all of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Mountain View, Calif.

[*] Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

[21] Appl. No.: **675,683**

[22] Filed: **Jul. 3, 1996**

[51] Int. Cl.⁶ **G06F 17/60**

[52] U.S. Cl. **705/26; 345/329; 345/333; 345/334; 345/335; 364/400; 395/200.33; 395/200.47; 395/200.49; 395/200.6; 705/35; 707/104**

[58] Field of Search **364/400, 464.1, 364/479.01, 479.02, 479.04, 479.06; 395/200.6, 200.33, 200.47, 200.48, 200.49, 201, 226, 227, 329, 330, 331, 333, 334, 335; 705/1, 26, 27, 400, 35; 345/329, 330, 331, 333, 334, 335; 707/104**

[56] **References Cited****U.S. PATENT DOCUMENTS**

3,314,049	4/1967	Felcheck	705/15
4,473,824	9/1984	Claytor	340/825.27
5,331,543	7/1994	Yajima et al.	705/31
5,357,439	10/1994	Matsuzaki et al.	364/468.02
5,570,291	10/1996	Dudle et al.	364/468.01
5,666,493	9/1997	Wojcik et al.	705/26
5,694,546	12/1997	Reisman	395/200.9
5,710,887	1/1998	Chelliah et al.	395/226

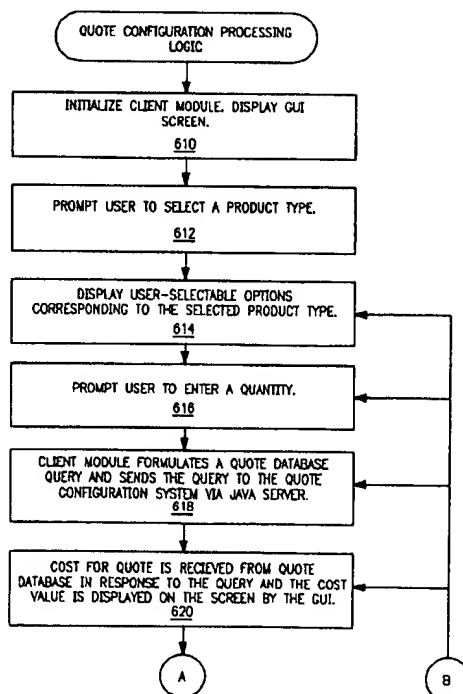
Primary Examiner—Edward R. Cosimano

Attorney, Agent, or Firm—Blakely Sokoloff Taylor & Zafman

[57] **ABSTRACT**

A platform-independent, usage-independent, location-independent quote configuration system is described. The present invention, operating in a computer network, is a quote configurator comprising, 1) a client module, the client module having a platform-independent user interface for receiving quote input and command selections from a user, the quote input and command selections including product selection and selection of information indicative of business rules, and 2) a server coupled to the client module across the network, the server having access to quote data and business rules, the server including a platform-independent server interface configured to receive the quote input and command selections from the client module, the server validating the quote input based on the quote data and the business rules.

24 Claims, 7 Drawing Sheets



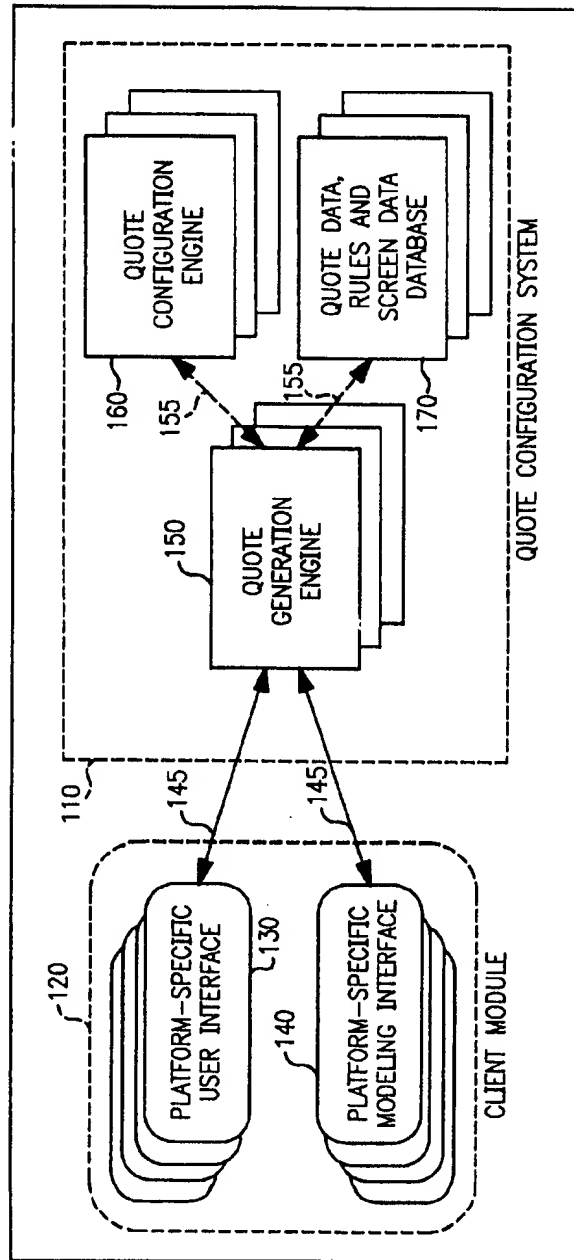


FIG. 1
PRIOR ART

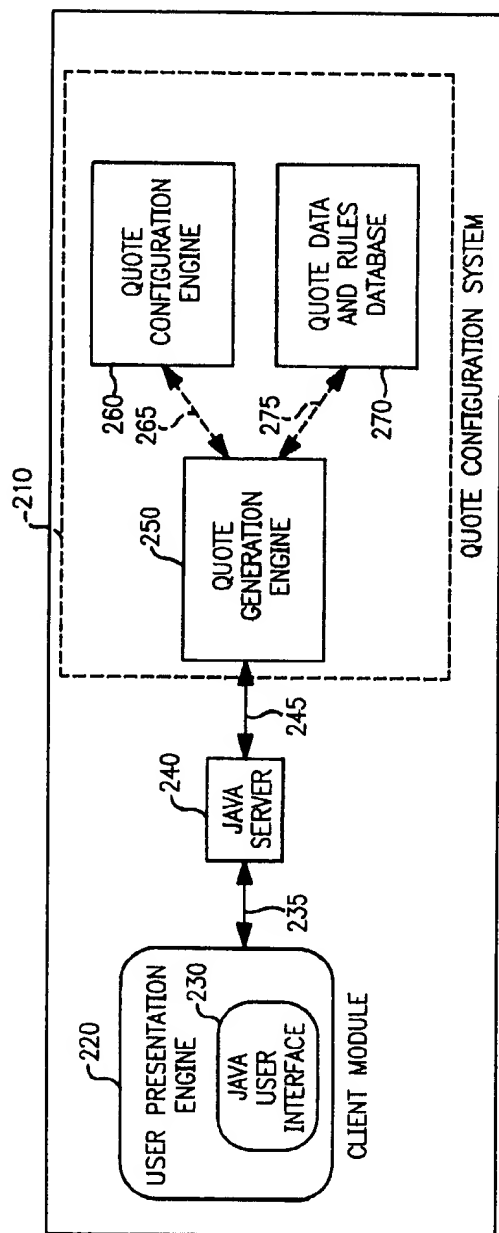


FIG. 2

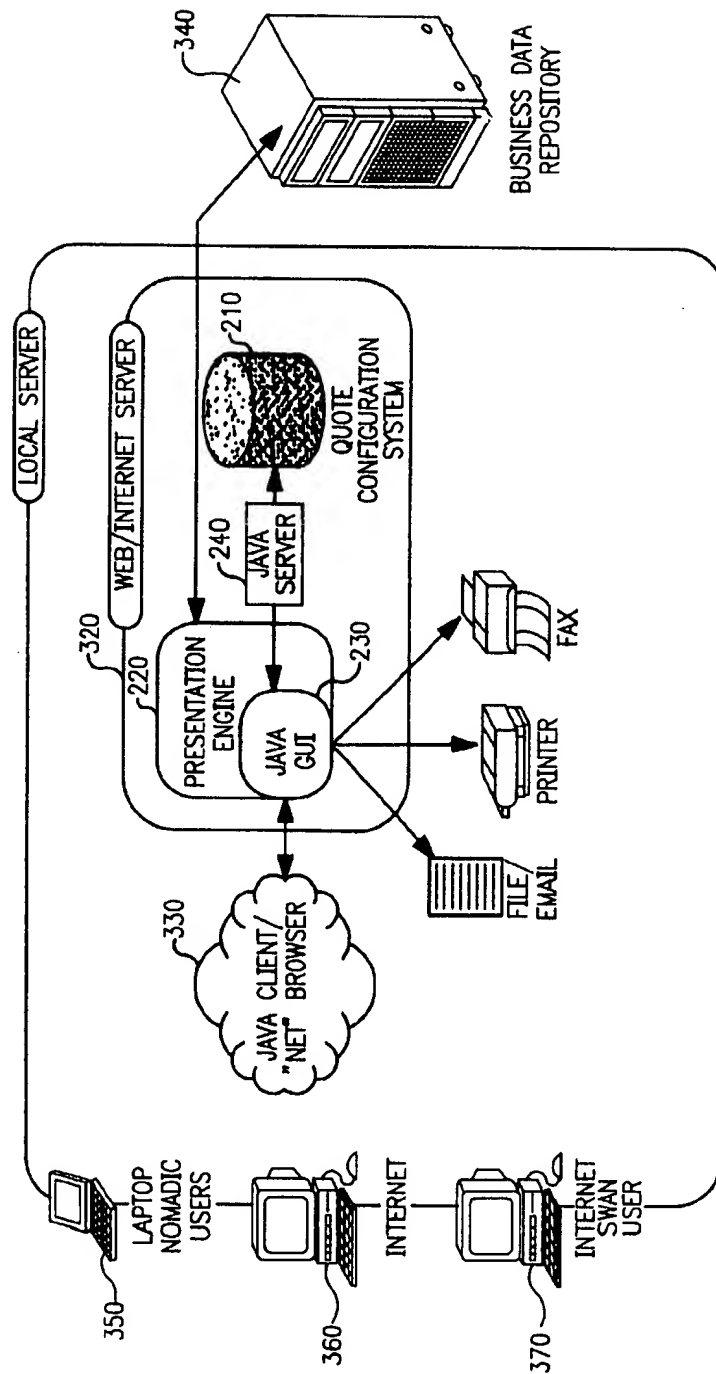


FIG. 3

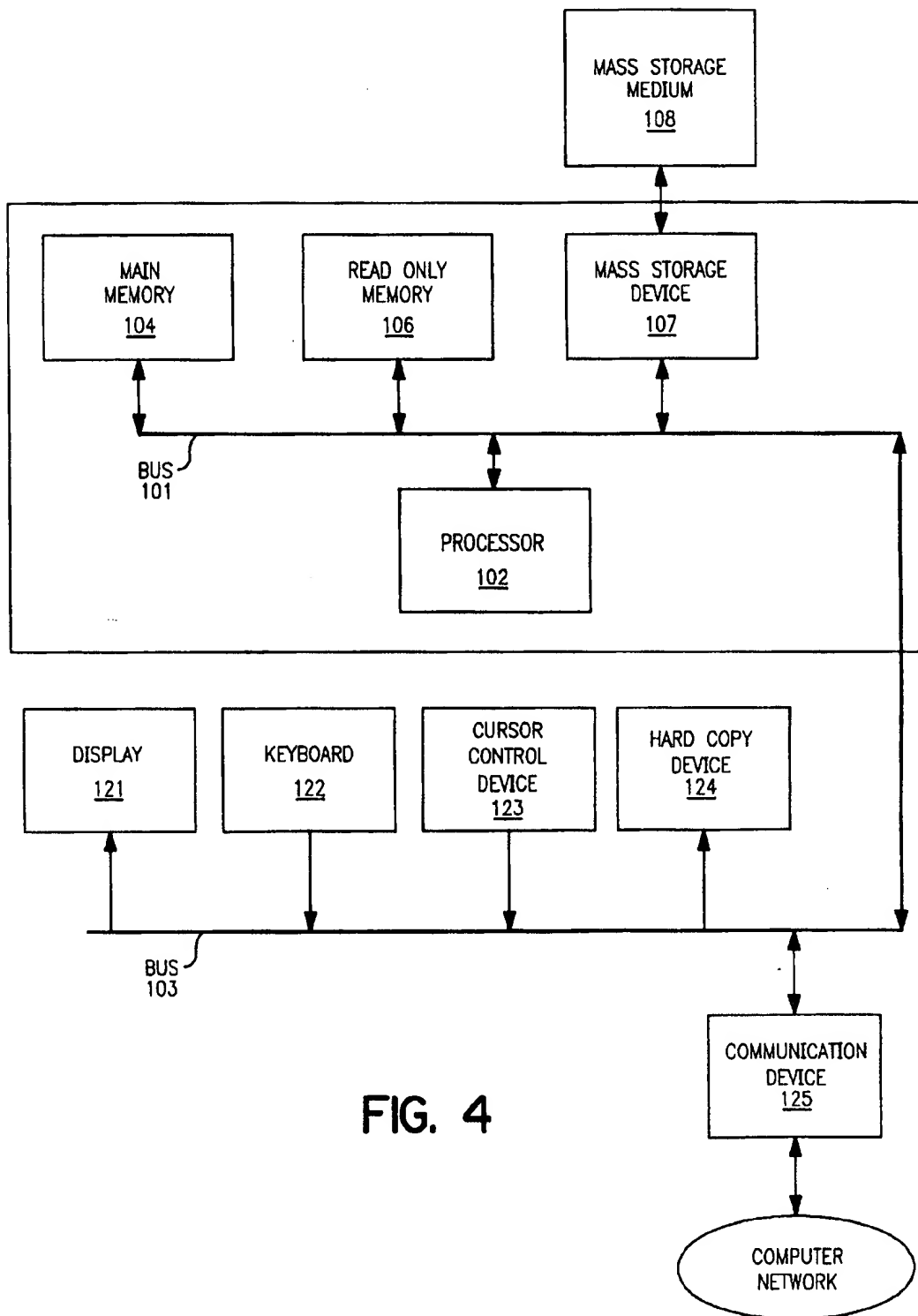


FIG. 4

File <input type="button" value="□"/>		Help <input type="button" value="□"/>			
Customer Name <input type="text"/>		Sales Agent <input type="text"/>			
Customer Address <input type="text"/>					
Products <input type="checkbox"/> S10 <input type="checkbox"/> S4/20 <input type="checkbox"/> 54/50		Product Selection Quantity			
Quantity <input type="text"/>	<input type="button" value="Add"/>	<input type="button" value="Change"/>	<input type="button" value="Delete"/>		
<div style="border: 1px solid black; height: 40px; width: 100%;"></div>					
Price List (Country): <input type="text" value="US"/>		Prog. type: <input type="checkbox"/> Global <input type="checkbox"/> Reseller <input type="checkbox"/> SchPAC			
Marketing Services: <input type="text" value="none"/>		Trav Zn: <input type="checkbox"/> 0-50 <input type="checkbox"/> 51-100 <input type="checkbox"/> 101-150 <input type="checkbox"/> 151-250			
Service Level: <input type="checkbox"/> Platinum <input type="checkbox"/> Gold <input type="checkbox"/> Silver <input type="checkbox"/> Bronze		<input type="button" value="View Service Detail"/>			
Options: Cust. Priority Phone Support On-site Resp. On-site Resp. On-site Resp.					
<input type="text" value="none"/>	<input type="text" value="none"/>	<input type="text" value="none"/>	<input type="text" value="no"/>	<input type="text" value="no"/>	
Discounts: Account End User Maint Mgr. Multi yr. Prepayment Monthly Price					
<input type="text" value="none"/>	<input type="text" value="none"/>	<input type="text" value="none"/>	<input type="text" value="none"/>	<input type="text" value="no"/>	<input type="text"/>
<input type="button" value="Prequote"/>	<input type="button" value="Connect"/>	<input type="button" value="Reset"/>	<input type="button" value="Exit"/>	<input type="button" value="Contradictions"/>	<input type="button" value="Quote"/>

FIG. 5

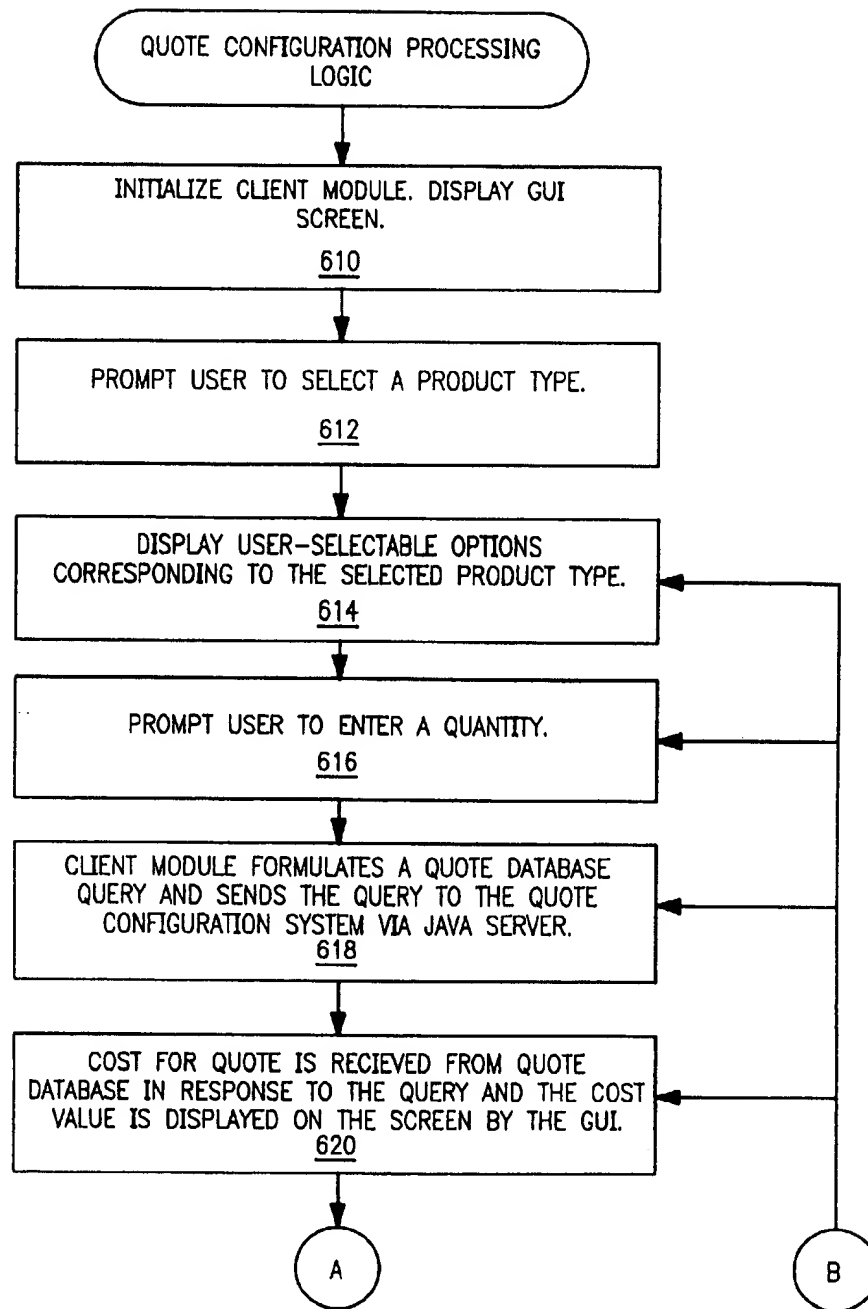


FIG. 6

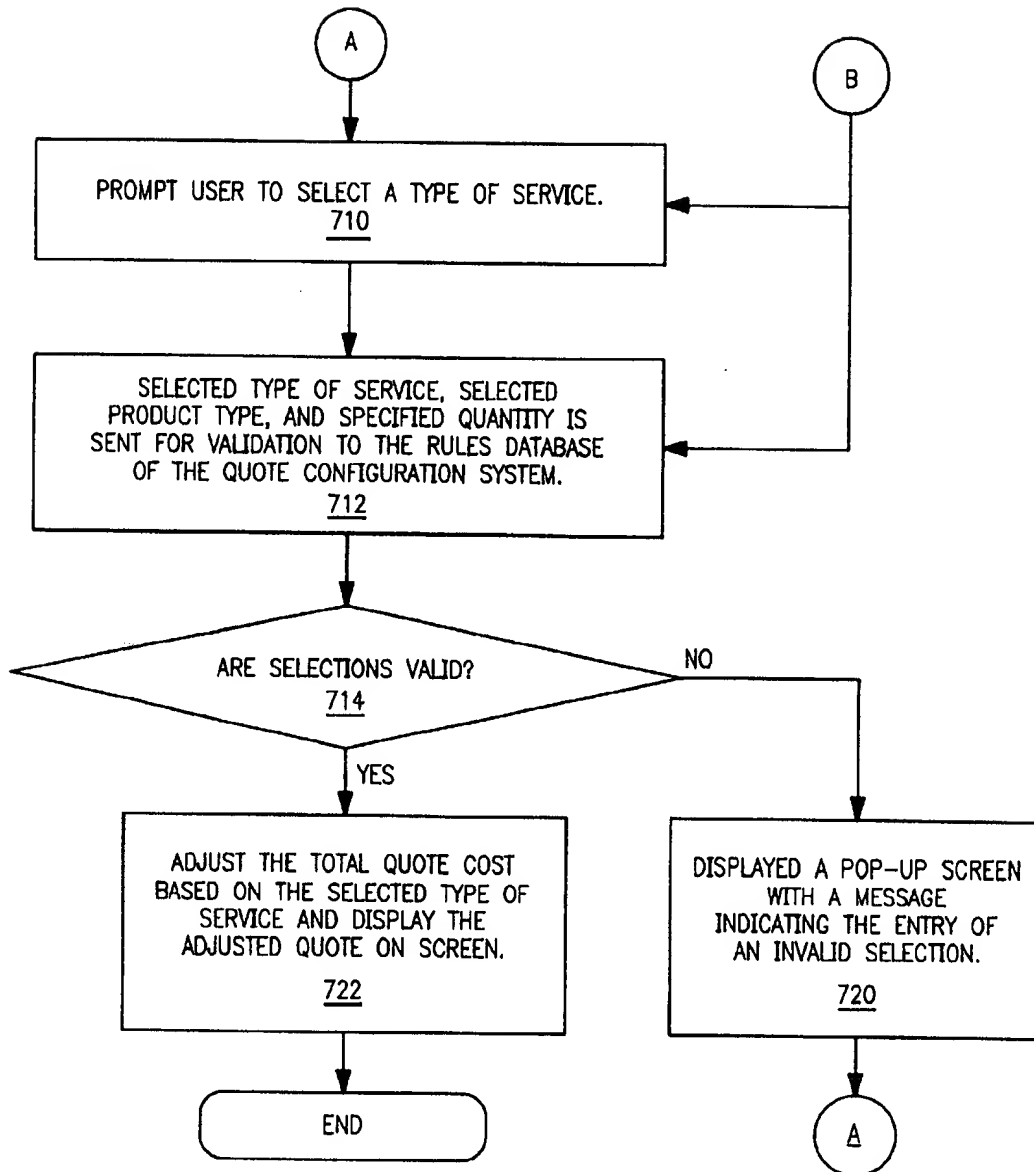


FIG. 7

1

PLATFORM-INDEPENDENT, USAGE-INDEPENDENT, AND ACCESS-INDEPENDENT DISTRIBUTED QUOTE CONFIGURATION SYSTEM

FIELD OF THE INVENTION

The present invention relates to the field of distributed computer systems. Specifically, the present invention relates to the field of distributed computer systems for configuring and providing pricing and quotation information regardless of usage or access model and independent of platform type.

BACKGROUND OF THE INVENTION

Pricing and quotation information is produced at different points and different geographic locations in the sales process. Some of the key triggers used to produce this information include product sales, volume discounts, warranty terms, sales/lease contract terms, contract renewals or modifications, or special customer requests. Often times, quotations or quotes produced as a result of these triggers vary substantially based on the business unit or geographic location for which a particular quote is targeted. These variations are also caused by the different computer platforms, systems, usage models and networks used to assemble the quote information. Variations can also be caused by lack of, or differing, in-depth knowledge of the specific business rules and discounts that apply to each scenario, location, and business unit. In some cases, duplicate and redundant information must be reformatted or translated to a form compatible with a component of the quote generation system. In other cases, information must be manually entered. This is due in large part to the rigid data input requirements of conventional quote configuration systems.

The limitations in current quote configuration systems result in quotes that are difficult to generate and maintain, which contain errors that impact the customer and the business units, and require manual entry of redundant information at each step of the process.

Thus, a platform-independent, usage-independent, location-independent quote configuration systems which is capable of encapsulating the business logic and business unit expertise to produce consistent, accurate results is needed.

SUMMARY OF THE INVENTION

A platform-independent, usage-independent, location-independent quote configuration system is described. The present invention, operating in a computer network, is a quote configurator comprising, 1) a client module, the client module having a platform-independent user interface for receiving quote input and command selections from a user, the quote input and command selections including product selection and selection of information indicative of business rules, and 2) a server coupled to the client module across the network, the server having access to quote data and business rules, the server including a platform-independent server interface configured to receive the quote input and command selections from the client module, the server validating the quote input based on the quote data and the business rules.

Thus, it is an advantage of the present invention over conventional systems that the present invention is based on a non-procedural data-driven work flow design. It is a further advantage of the present invention that the point of service provides a real-time, interactive model for interaction on all components with the user. It is a further advantage

2

of the present invention that all business rules and expertise needed to provide intelligent quoting are encapsulated within the quote configuration system, and are transferable and usable anywhere on any platform. This enables the design to be scalable with localization possible to accommodate variations in attributes such as currency and price list. It is a further advantage of the present invention that the present invention is based on an open architecture to enable integration with other systems, business processes and technologies. It is a further advantage of the present invention that the present invention is platform-independent (ex. SPARC™, RISC, X86). It is a further advantage of the present invention that the present invention is independent of usage model (ex. nomadic, remote/dial-up, inter-networked, intra-networked). It is a further advantage of the present invention that the present invention is independent of access model (ex. stand-alone application, networked application, web-based application, web-based applet).

The features and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description of the preferred embodiment of the present invention as set forth below.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the accompanying drawings, in which:

FIG. 1 illustrates the system architecture of a prior art quote configuration system.

FIG. 2 illustrates the architecture of the quote configurator of the preferred embodiment of the present invention.

FIG. 3 illustrates the operation of the quote configurator of the preferred embodiment of the present invention.

FIG. 4 illustrates a typical data processing system or platform upon which one embodiment of the present invention is implemented.

FIG. 5 illustrates a sample screen display of the graphical user interface of the preferred embodiment.

FIGS. 6 and 7 are flowcharts illustrating the operation of the preferred embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention is a platform-independent, usage-independent, location-independent quote configuration system. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one of ordinary skill in the art that these specific details need not be used to practice the present invention. In other instances, well-known structures, interfaces and processes have not been shown in detail in order not to unnecessarily obscure the present invention.

FIG. 4 illustrates a typical data processing system or platform upon which one embodiment of the present invention is implemented. It will be apparent to those of ordinary skill in the art, however that other alternative systems of various system architectures may also be used. The data processing system illustrated in FIG. 4 includes a bus or other internal communication means 101 for communicating information, and a processor 102 coupled to the bus 101 for processing information. The system further comprises a random access memory (RAM) or other volatile storage device 104 (referred to as main memory), coupled to bus 101 for storing information and instructions to be executed by processor 102. Main memory 104 also may be used for

storing temporary variables or other intermediate information during execution of instructions by processor 102. The system also comprises a read only memory (ROM) and/or static storage device 106 coupled to bus 101 for storing static information and instructions for processor 102, and a data mass storage device 107 such as a magnetic disk drive or optical disk drive. Data storage device 107 is coupled to bus 101 and is typically used with a computer readable mass storage medium 108, such as a magnetic or optical disk, for storage of information and instructions. The system may further be coupled to a display device 121, such as a cathode ray tube (CRT) or a liquid crystal display (LCD) coupled to bus 101 through bus 103 for displaying information to a computer user. An alphanumeric input device 122, including alphanumeric and other keys, may also be coupled to bus 101 through bus 103 for communicating information and command selections to processor 102. An additional user input device is cursor control 123, such as a mouse, a trackball, stylus, or cursor direction keys coupled to bus 101 through bus 103 for communicating direction information and command selections to processor 102, and for controlling cursor movement on display device 121. Another device which may optionally be coupled to bus 101 through bus 103 is a hard copy device 124 which may be used for printing instructions, data, or other information on a medium such as paper, film, or similar types of media. In the preferred embodiment, a communication device 125 is coupled to bus 101 through bus 103 for use in accessing other nodes of a network computer system. This communication device 125 may include any of a number of commercially available networking peripheral devices such as those used for coupling to an Ethernet, token ring, Internet, or wide area network.

Note that any or all of the components of the system illustrated in FIG. 4 and associated hardware may be used in various embodiments of the present invention; however, it will be appreciated by those of ordinary skill in the art that any configuration of the system may be used for various purposes according to the particular implementation. In one embodiment of the present invention, the data processing system illustrated in FIG. 4 is an IBM® compatible personal computer (PC) or a SUN® SPARC™ Workstation. Processor 102 may be one of the 80X86 compatible microprocessors such as the 80486 or PENTIUM® brand microprocessors manufactured by INTEL® Corporation of Santa Clara, Calif.

The software implementing the present invention can be stored in main memory 104, mass storage device 107, or other storage medium locally accessible to processor 102. It will be apparent to those of ordinary skill in the art that the methods and processes described herein can be implemented as software stored in main memory 104 or read only memory 106 and executed by processor 102. This software may also be resident on an article of manufacture comprising a computer usable mass storage medium 108 having computer readable program code embodied therein and being readable by the mass storage device 107 and for causing the processor 102 to perform quote transactions and protocols in accordance with the teachings herein.

The preferred embodiment of the present invention is a platform-independent, usage-independent, location-independent quote configuration system, which can be implemented on computer system platforms such as the one illustrated in FIG. 4.

ARCHITECTURE OF THE PREFERRED EMBODIMENT

FIG. 1 illustrates the typical quote configuration generation system currently used in the prior art. The architecture

of a typical conventional quote processing system is based on a two-tier procedural model with deliberate data synchronization latency. The client module 120 is typically large, requiring significant local client resources (such as disk space, memory, swap space, etc.) to run the quote configuration system. The client module 120 includes a platform-specific modeling interface 140 and a platform-specific user interface 130. The client module 120 is coupled to a conventional quote configuration system 110 across a network connection using a transaction messaging mechanism 145, which is compatible with a conventional quote generation engine 150 in quote configuration system 110. Quote generation engine 150 is then coupled to quote configuration engine 160 and quote database 170 across an interface using an intrasystem communication protocol 155. Quote database 170 provides the quote data, rules, and screen data used by the quote generation engine 150 along with quote configuration engine 160 to generate a quote. Multiple configuration system databases are deployed to accommodate location-specific variables.

This prior art quote configuration methodology has several disadvantages, which are listed below:

1. A different platform-specific user interface 130 and client module 120 is necessary for each type of hardware architecture and operating system (OS) (i.e. platform) upon which the quote configuration system is deployed.

2. A different platform-specific user interface 130, client module 120, transaction messaging mechanism 145, and intrasystem communication protocol 155 is necessary for each type of usage model for which the quote configuration system is implemented. In conventional systems, the four usage models commonly utilized are nomadic, remote/dial-up, inter-networked and intra-networked. In a nomadic system, the client module is resident in a portable computer system, which only periodically and asynchronously connects to the quote configuration system. In a remote/dial-up model, the client module uses a dial-up modem protocol to asynchronously connect with the quote configuration system 110. In an inter-networked model, the client module uses Internet or World Wide Web (WWW) protocols to synchronously but indirectly connect to a quote configuration system. In an intra-networked model, the client module uses local area network (LAN) or wide-area network (WAN) protocols to synchronously and directly connect to a quote configuration system.

3. A different platform-specific user interface 130, client module 120, transaction messaging mechanism and intrasystem communication protocol is necessary for each type of access model for which the quote configuration system is implemented. The four access models commonly provided in conventional systems are stand-alone application, networked application, web-based application and web-based applet. Stand-alone applications are those software systems that run on a platform without network connectivity. Networked applications are those software systems that run on a platform with network connectivity and involve network transactions. The web-based models run on the World Wide Web (WWW) portion of the Internet and have dependencies on browser-specific implementations in addition to platform-specific dependencies. All four of these prior art access models have user interface screen rendering logic and data which is tightly integrated with the business rules and quote data in the quote configuration system 110.

4. The system architecture and application programming interfaces (APIs) are proprietary and closed.

5. The work flow design is strictly procedural in a batch style of processing and has deliberate latency built into the data synchronization model.

6. Distribution of the quote configuration system and client software, synchronization of quote data and business rules, and version control of the system are difficult to manage effectively.

7. A different quote configuration engine 160 and quote database 170 must be deployed to accommodate each location-specific environment.

8. The user interface 130 requires extensive knowledge of the underlying business rules and quote data to generate a configuration quote.

9. The prior art design is based on a two-tiered, non-distributed/non-networked model.

FIG. 2 illustrates the quote configurator of the preferred embodiment of the present invention. In the present invention all common configurations and usage models are enabled by a single architecture and single user interface model, including mobile computing devices (nomadics and remote access/dial-ups), local area networks (LANs), wide area networks (WANs) and the Internet network. In general, these components provide specific functionality for handling quotation information and for generating quotes.

The architecture of the present invention is based on a distributed three-tier object-oriented design model as shown in FIG. 2. A first tier is a client module 220. The second tier is a server 240 and the third tier is quote configuration system 210.

The client module 220 includes a JAVA™ user interface 230. JAVA is an object-oriented programming language similar to C++ developed by SUN® Microsystems of Mountain View, Calif. JAVA is unique in that a program written in JAVA is first compiled into byte-codes and subsequently interpreted into machine dependent processor instructions. Byte-codes are similar to machine instructions; however, byte-codes are not specific to a particular machine or platform. Thus, JAVA programs can run on any platform that supports JAVA. It is not necessary to recompile a JAVA program to run on a new machine. The JAVA user interface 230 may therefore be written without platform-specific code and without the need to create multiple versions that support multiple platforms.

Each tier of the system illustrated in FIG. 2 encapsulates all the necessary components for the intended function. All interfaces of the present invention are open and public. The messaging transport system of the present invention is based on socket-level communication protocol using open APIs at each end of the communications path, such as paths 235 and 245. Socket-level communication protocol in the context of a computer network is well known to those of ordinary skill in the art. The user presentation engine of client module 220, user interface 230, and JAVA server 240 components are platform-independent, usage-independent, and access-independent and based on a single code source. The JAVA server 240 API which interfaces path 245 to quote generation engine 250 is an essential element of the present invention. Similarly, the JAVA server 240 API which interfaces path 235 to client module 220 is also an essential element of the present invention. Specifically, the user interface component 230 is a graphical user interface (GUI) and looks and operates in exactly the same manner independently of platform, usage, and access model. The GUI 230 is fully interactive and intelligent, and requires absolutely no knowledge of the underlying business rules and quote data for a user to generate an accurate configuration quote. The interaction of the GUI 230 is field-based, data-driven and event-driven. The quote data and rules database 270 also contains new components to support the present invention.

The client module 220 API to quote generation engine 250 through JAVA server 240 comprises commands, which are formulated in a platform-independent text string format and sent to quote generation engine 250 via a socket interface of JAVA server 240. A specified port number is used by both the client module 220 and the quote generation engine 250. The quote generation engine 250 interprets these commands and sends back its replies to the client module 220 via the socket interface of JAVA server 240. A detailed description of the commands of this API follows.

Commands

bload rtssun7b.cab

This command makes the quote generation engine 250 read and load (in memory) all information contained in the rtssun7b.cab file. This file contains all business rules and screen rendering information needed by the quote generation engine 250.

BEruntimeInit streamout

This command initiates runtime a communication protocol by the quote generation engine 250. The 'streamout' argument is a parameter that tells the quote generation engine 250 to use its socket layer interface for communication.

BEinsPickByIndex <field> <idx> 1

This command instructs the quote generation engine 250 to pick a <field> with index <idx> and return all information stored for this field. The <field> argument can be any of the following in the preferred embodiment:

Contract	PriceBook	MarketingServices
ReplcHrdwrPrt	CstDefPrty	UnlimPhoneSprt
OnSiteResp	OnSiteResp2_hr	OnSiteCov7_24
PrefCustomer	EndUser	MaintMan
Multiyr		

The <idx> argument is an integer value representing the index of the specified field.

Example: BEinsPickByIndex Contract 2 1

(This sample command requests the quote generation engine 250 to retrieve all the information about the second field for contracts.)

BEinsValueSet NumberUsers val val <qty>

This command sets the quantity of products specified by the user. The <qty> argument is an integer value that represents the quantity of products.

BEReset

This command forces the quote generation engine 250 to ignore all previous input and start from the very beginning.

BEprequote

This command produces a pre-quote listing for the current session.

ueQuote

This command produces a final quote for the current session.

BEinsWhy <field> <state>

This command returns an explanation as to why a state of contradiction now exists for this <field>.

Example: BEinsWhy OnSiteResp2_hr 2 (This example asks why is there a contradiction in 2-hr response time?)

BEviewSummaryGet <windowname>

This command returns comprehensive information about a set of fields that logically belong to <windowname>.

Example: BEviewSummaryGet win4 (The quote generation engine 250 returns all current information and states of fields that belong to window 4.)

These interfaces and commands of the preferred embodiment are only a subset of the entire API; however, these commands are sufficient for an operable system. It will be apparent to those of ordinary skill in the art that other commands or command arguments may also be provided. Conversely, the interfaces 265 and 275 to the quote configuration engine 260 and quote database 270, respectively, are conventional interfaces.

OPERATION OF THE PREFERRED EMBODIMENT

FIG. 3 illustrates the operation of the quote configurator of the preferred embodiment of the present invention. The client module 220 communicates with JAVA server 240 and quote configuration system 210 in the context of the World Wide Web (WWW) and the Internet 320. The client module 220 may also gather other business data from repository 340. Network users of the present invention such as nomadic users 350, Internet users 360, or intranet users 370 may access the client module 220 of the present invention via JAVA client/net browser 330. Once these network users connect to the client module 220, the users may request and view quote information. Quote information may also be printed, faxed, filed, or emailed via output devices connected to client module 220.

The following sample scenario will demonstrate an example of the quote configuration process of the present invention for generating a custom quote from both the user and system perspective. This example will highlight the essential steps of the present invention as appropriate. In this description, no distinction is made regarding location of the user/system, access method of the user/system, or the type of platform on which either the user or system exists. The same process and model will work regardless of platform, access type, or usage model. FIGS. 2 and 3 are referenced in combination with the flow diagrams of FIGS. 6 and 7 to illustrate the operation of the present invention. In addition, the screen sample shown in FIG. 5 also referenced.

On initiation of the client module 220 of the present invention by the user, the GUI 230 establishes a socket connection with the JAVA server 240 in real-time, and loads an initial data set. In block 610, shown in FIG. 6, a main screen or window of the GUI 230 is then auto-populated with a series of default values as specified by the JAVA GUI 230, based on the encapsulated business rules. One such sample screen is illustrated in FIG. 5. These default values include Price Lists; Travel Zone; Service Level. At this point in the process, the user only needs to select a product type and enter a quantity in order to receive a quote. In this example, the user will be producing a custom quote.

1. User selects the product.

The user is prompted in block 612, shown in FIG. 6, with a pick list to select a product type or the user can ask for additional information/help. On selection of the product type from the pick list presented by the GUI 230, the GUI 230 logic updates the screen with appropriate user-selectable options, key hints and other information intended to assist the user with creation of the custom quote based on the user selection of product type (block 614).

2. User enters a quantity.

On entry of the quantity (block 616), the GUI 230 logic determines the initial total cost for the quote by using the JAVA server 240 and socket protocol via path 245 to query the quote database 270 (block 618), perform the calculations and update the appropriate portion of the screen of the GUI 230 (block 620). As denoted by bubble B shown in FIGS. 6

and 7, the user may iteratively select or edit the selections described in blocks 614-712.

3. User selects the types of coverage desired.

Referring now to FIG. 7, selection of the type of coverage provides an adjustment in total quote cost based on the type service selected (blocks 710 and 712 shown in FIG. 7). This selection of type of coverage identifies a set of business rules which is applied to the request. If this type of coverage is not available for the selected product type and the specified quantity, the business rule of the quote configuration system quote database will indicate an invalid condition which is communicated to the GUI 230 via JAVA server 240 (block 714). This invalid condition causes a pop-up screen to appear for the user (block 720). The pop-up screen contains information and navigation/event assistance for the user. These pop-up screens indicating invalid conditions, warnings and hints can be displayed at any time and triggered by any appropriate event; because, the JAVA server 240 and quote configuration system 210 are in frequent communication regarding the state of the GUI 230 and selected events/options. Also, the user can "override" the suggestion/warning pop-up screen message and proceed with the quote anyway in order to accommodate local variations/conditions.

In addition to event-driven assistance, the user can get on-demand assistance on invalid conditions, additional hints and warnings by selecting the "Contradictions" button as shown on the screen sample shown in FIG. 5. After the quote is adjusted based on the selected product type, the specified quantity, and type of service, the adjusted quote is displayed for the user on the screen (block 722). The user can view and process the quote at any time during the configuration process.

The user has the following additional options provided at any time during the quote process in the preferred embodiment: 1) view the quote on-line; 2) fax the quote to a fax-enabled destination using a conventional facsimile device; 3) e-mail the quote to a networked destination using a conventional electronic mail system; 4) print the quote to a system-acknowledged printer; or 5) save the quote to a file. Buttons for selecting these options are shown in FIG. 5 and the interface supporting these options is shown in FIG. 3.

Thus, a platform-independent, usage-independent, and access-independent quote configuration system has been described. The specific arrangements and methods described herein are merely illustrative of the principles of the present invention. Numerous modifications in form and detail may be made by those of ordinary skill in the art without departing from the scope of the present invention. Although this invention has been shown in relation to a particular embodiment, it should not be considered so limited. Rather, the present invention is limited only by the scope of the following claims.

We claim:

1. In a computer network, a quote configurator comprising:
 - a client module, the client module having a user interface to receive quote input and command selections from a user, the quote input and command selections including product selection and selection of information indicative of business rules; and
 - a server coupled to the client module across the network, the server having access to quote data and business rules, the server including a server interface configured to receive the quote input and command selections from the client module, the server validating the quote input based on the quote data and the business rules,

- said user interface and said server interface being comprised of platform independent byte-codes.
2. The quote configurator as claimed in claim 1 wherein the server interface is a platform-independent, usage-independent and access-independent transport protocol for messaging quote data and business rules between the server and the client module.
 3. The quote configurator as claimed in claim 1 wherein the user interface is a platform-independent, usage-independent and access-independent graphical user interface.
 4. The quote configurator as claimed in claim 1 wherein the client module is JAVA compatible.
 5. The quote configurator as claimed in claim 1 wherein the server is JAVA compatible.
 6. The quote configurator as claimed in claim 1 wherein the client interface byte-codes are interpreted for operation on a particular client platform.
 7. The quote configurator as claimed in claim 1 wherein the server interface byte-codes are interpreted for operation on a particular server platform.
 8. The quote configurator as claimed in claim 1 wherein the user interface includes a screen display configured for product selection and selection of information indicative of business rules.
 9. The quote configurator as claimed in claim 1 wherein the client module further includes an interface to a nomadic user.
 10. The quote configurator as claimed in claim 1 wherein the client module further includes an interface to an Internet user.
 11. The quote configurator as claimed in claim 1 wherein the client module further includes an interface to an intranet user.
 12. A method for configuring quotes comprising:
 - receiving quote input and command selections from a user via a user interface in a client module, the quote input and command selections including product selection and selection of information indicative of business rules;
 - sending said quote input and command selections to a server across a server interface, said user interface and server interface being comprised of platform independent byte-codes;
 - accessing quote data and business rules; and
 - validating the quote input based on the quote data and the business rules.
 13. The method as claimed in claim 12 wherein the sending further includes preparing the quote input and command selections in a platform-independent, usage-independent and access-independent transport protocol for transport to the server.
 14. The method as claimed in claim 12 wherein the user interface is a platform-independent, usage-independent and access-independent graphical user interface.

15. The method as claimed in claim 12 wherein the client module is JAVA compatible.
16. The method as claimed in claim 12 wherein the server is JAVA compatible.
17. The method as claimed in claim 12 further including interpreting the client interface byte-codes for operation on a particular client platform.
18. The method as claimed in claim 12 further including interpreting the server interface byte-codes for operation on a particular server platform.
19. The method as claimed in claim 12 further including receiving a product selection and receiving a selection of information indicative of business rules.
20. The method as claimed in claim 12 further including receiving input from a nomadic user.
21. The method as claimed in claim 12 further including receiving input from an Internet user.
22. The method as claimed in claim 12 further including receiving input from an intranet user.
23. On a computer useable medium having computer readable code embodied therein, a client module for causing the computer to configure quotes, the client module comprising:
 - computer readable program code configured to cause the computer to receive quote input and command selections from a user via a user interface, the quote input and command selections including product selection and selection of information indicative of business rules;
 - computer readable program code configured to cause the computer to send said quote input and command selections to a remote server across a server interface; and
 - computer readable program code configured to cause the computer to receive validation information from the remote server indicating whether the quote input is valid based on quote data and the business rules,
- said user interface and said server interface being comprised of platform independent byte-codes.
24. On a computer useable medium having computer readable code embodied therein, a server for causing the computer to configure quotes, the server comprising:
 - computer readable program code configured to cause the computer to receive quote input and command selections from a remote client module across a server interface;
 - computer readable program code configured to cause the computer to access quote data and business rules; and
 - computer readable program code configured to cause the computer to validate the quote input based on the quote data and the business rules,
- said server interface being comprised of platform independent byte-codes.

* * * * *